Colorisation d'image par transfert de couleur

Melvin Even

Abstract—Le but de ce projet est d'implémenter la méthode de colorisation proposée dans l'article de Welsh et al. [1] qui permet de transférer une palette de couleurs d'une image référence à une image en niveau de gris. L'utilisateur peut aussi guider le procédé en faisant correspondre des régions des deux images.

1 Introduction

La colorisation d'image en noir et blanc est souvent utilisé pour améliorer l'attrait visuel d'images et films anciens. De plus, l'ajout de couleurs à certaines images et illustrations scientifiques permet d'augmenter la compréhension du contenu en utilisant par exemple des variations de couleur.
Le problème de colorisation d'image peut être défini ainsi : étant donné une image en niveau de gris, c'est-à-dire une distribution d'intensité de couleur (luminance), affecter à chaque élément (pixel) une valeur chromatique.
Pour une valeur de luminance donnée, il est possible de lui associer une infinité de valeur chromatique. Décider de la validité d'un résultat est alors basé principalement sur la perception humaine. Il est généralement nécessaire d'avoir l'appui d'un humain pour le choix de palette de couleurs, ce qui est d'autant plus vrai lorsque la colorisation à des fins artistiques, par exemple pour la création d'illustrations scientifiques.

Ce travail de recherche se base sur l'article de Welsh et al. [1]. Ils proposent une méthode de colorisation qui consiste à transférer une palette de couleurs d'une image de référénce, choisie par l'utilisateur, vers une image en niveau de gris. L'utilisateur a aussi la possibilité de guider le procédé en sélectionnant des régions des deux images à mettre en correspondance.

1.1 Travaux connexes

La plupart des premières techniques de colorisation de films sont propriétaires et il existe très peu de documentation dessus. Un article de 1998 dans CineSite évoque une méthode qui consiste à segmenter des images pour qu'un humain puisse colorier les segments un à un. Dans le cas d'un film ou d'une vidéo, un algorithme suit les mouvements des différents segments dans chaque image de la vidéo et propage les couleurs des segments dans le temps.

La pseudo-colorisation [2] permet de coloriser une image avec une table de couleurs arbitraire. Plus formellement, elle peut être décrite comme une transformation T telle que $c_{x,y} = T(g_{x,y})$ où $c_{x,y}$ et $g_{x,y}$ sont respectivement les pixels en couleur et en niveau de gris. Cette technique est souvent utilisée pour coloriser des images scientifiques. Par exemple, dans le cas d'une image d'IRM pseudo-colorisée, il y a une correspondance entre chaque valeur de densité et des couleurs arbitraires.

L'article de Welsh et al. [1] est basé sur le travail de Reinhard et al.[3] qui propose une méthode pour transférer une palette de couleurs entre deux images de couleurs. Le principe est de trouver des correspondances entre les pixels selon la distribution des couleurs des images, puis d'altérer la distribution des couleurs de l'image cible pour qu'elle se rapproche de celle de l'image source.
Levin et al. [4] proposent une méthode où l'utilisateur dessine des traits de couleurs sur l'image. Les couleurs sont ensuite propagées dans l'espace et le temps (pour une séquence d'image). Cette approche reformule le problème en un problème d'optimisation et ne requiert pas de segmentation précise de l'image.

Plus récemment, les nouvelles techniques de colorisation se basent principalement sur l'utilisation de réseaux de neurones convolutifs [5] [6] et de GAN [7]. Ces méthodes ne requièrent pas d'intervention humaine et peuvent être complètement automatisées. L'idée est de prédire la couleur d'un pixel en niveau de gris en se basant sur la distribution de luminance de l'image. Certaines méthodes adoptent en plus une approche d'analyse sémantique de l'image. Les différents éléments trouvés sont alors colorisés à l'aide d'une base de données de couleurs [8].

2 Approche et méthode

2.1 Définitions

Nous clarifions ici quelques termes utilisés dans les parties suivantes.
La **luminance** d'un pixel correspondant à l'intensité de la luminosité. Elle est indépendante de la chrominance (couleur).
Une **texture** est un ensemble de pixel caractérisé par la variation de leur luminance.
Le **voisinage** d'un pixel est une fenêtre de taille fixe centrée sur le pixel.

2.2 Transfert de couleur

L'algorithme de transfert de couleur prend entrée une image de référence en couleur ainsi que l'image en niveau de gris. L'idée est de trouver pour chaque pixel dans l'image en niveau de gris un pixel correspondant dans l'image en couleur. Pour cela, nous partons du principe qu'il y a une forte chance que deux pixels faisant partie d'une même texture aient une couleur similaire.
La texture d'un pixel est définie ici par la variation de luminance de son voisinage.
Les étapes principales de l'algorithme sont :

1) Transformer l'espace de couleur des deux images vers l'espace de couleur $l\alpha\beta$.
2) Rapprocher les histogrammes de luminance des deux images.
3) Échantillonner des pixels de l'image en couleur.
4) Faire correspondre chaque pixel dans l'image en niveau de gris à un des échantillons de l'image de référence, puis transférer sa couleur.
5) Transformer l'image résultante dans l'espace de couleur d'origine.

2.2.1 Changement d'espace de couleur

La première étape est de transposer les deux images dans l'espace de couleur $l\alpha\beta$ [9] [10]. L'avantage de cet espace de couleur est qu'il minimise la corrélation entre les différentes dimensions de l'espace. Cela a pour effet de réduire l'occurance d'artefacts visuels lorsqu'on applique des opérations de couleur sur l'image. Dans cet espace, la dimension l représente la luminance, α représente l'intervalle de couleur rouge-vert et β l'intervalle jaune-bleu. L'espace $l\alpha\beta$ est défini grâce à un ensemble d'expériences basées sur la perception humaine, et sur le principe que deux couleurs perçues par nos cônes ne peuvent pas être rouge et verte en même temps ou jaune et bleue en même temps.
La transformation de l'espace RGB en $l\alpha\beta$ passe d'abord par la transformation de RGB en LMS [11]. LMS est un espace de couleur basé sur la sensibilité des cônes de l'oeil humain. Cette transposition correspond à un ensemble de transformations linéaires de l'espace RGB vers l'espace cible.

2.2.2 Histogramme de luminance

La correspondance entre les pixels de l'image en niveau de gris et celle en couleur est basée sur leur luminance. En particulier, elle est basée sur des statistiques de la distribution de luminance du voisinage des pixels. En revanche, même si deux images possèdent des textures similaires, la correspondance peut être faussée à cause des conditions d'illumination globale de la scène. Par exemple, il est difficile de faire correspondre une même texture provenant d'une photographie prise en pleine journée et une prise au crépuscule.
Afin de prendre en compte ces différences de luminance globale sur l'image, nous appliquons une phase de *remapping* de luminance, proposée par Hertzmann et al. [12] sur l'image en couleur. Le principe est de modifier l'histogramme de luminance de l'image de référence pour qu'il se rapproche de celui de l'image en niveau de gris. L'histogramme de luminance d'une image représente la distribution des valeurs de luminance de chaque pixel (voir Fig. 1).
Le *remapping* se fait en appliquant une translation et une mise à l'échelle sur l'histogramme de l'image en couleur. Si on considère $Y(p)$ la luminance d'un pixel p dans l'image en couleur, la fonction de correspondance est

$$Y(p) = \frac{\theta_G}{\theta_C}(Y(p) - \mu_C) + \mu_G$$

où θ_G et θ_C représentent l'écart-type des luminances et μ_G et μ_C les luminances moyennes dans les images en niveau de gris et en couleur respectivement. Le *remapping* aide à faire correspondre les intervalles de luminance des deux images sans altérer la distribution de l'image en niveau de gris.

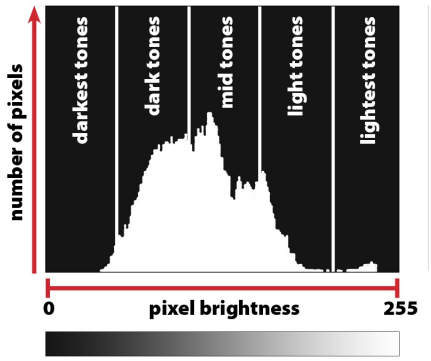


Fig. 1: Exemple d'un histogramme de luminance. Source: creative-photographer.com

2.2.3 Échantillonnage de l'image en couleur

Le but principal de l'algorithme est de faire correspondre chaque pixel de l'image en niveau de gris avec un pixel de l'image en couleur. Pour cela, on compare des statistiques du voisinage d'un pixel en niveau de gris avec celles d'un ensemble d'échantillons de pixels dans l'image en couleur. On peut d'abord utiliser l'ensemble des pixels de l'image en couleur comme échantillon, c'est-à-dire qu'on compare chaque pixel d'une image avec chaque pixel de l'autre. Les résultats montrent qu'utiliser l'ensemble de l'image comme échantillon de comparaison n'améliore pas la qualité du résultat et peut dans certains cas produire de fausses corrélations à cause d'un trop grand nombre d'échantillon.

Comme la perception de changement de l'apparence d'éléments d'une image est principalement due à la variation de luminance, on peut réduire le nombre d'échantillons en gardant un ensemble de couleur diversifié et représentatif. Par l'expérience, nous trouvons qu'en prenant un ensemble d'échantillons de taille fixe, environ 200, nous obtenons des résultats satisfaisants et pour des performances beaucoup plus acceptables. Pour échantillonner l'image, on la divise en un certain nombre de régions rectangulaires, puis on pioche aléatoire et uniformément un nombre fixe de pixels dans chaque région.

2.2.4 Correspondance entre pixels et transfert de couleur

L'idée derrière l'étape de correspondance entre pixels est de trouver le pixel de l'image en couleur le plus susceptible de partager la même texture qu'un pixel dans l'image en niveau de gris. On défini comme texture d'un pixel une fenêtre de taille fixe, dans notre cas 5x5, centrée sur le pixel considéré. L'apparence d'une texture est caractérisé par la variation de luminance des différents pixels qui la compose. Pour tester la similitude entre deux textures, il est alors naturel de comparer des statistiques sur la distribution de luminance.
Ici, on associe à chaque pixel p les valeurs μ_v, θ_v où μ_v est la moyenne pondérée et θ_v est l'écart-type des luminances du voisinage de p.

On précalcule d'abord les valeurs μ_v, θ_v des échantillons de couleur. Ensuite, on calcule μ_v et θ_v pour chaque pixel de

l'image en niveau de gris. La correspondance entre un pixel gris p_G et un échantillon p_C se fait en minimisant la valeur

$$D(p_C, p_G) = \frac{1}{2}(\mu_C - \mu_G)^2 + \frac{1}{2}(\theta_C - \theta_G)^2$$

où μ_C et θ_C sont les statistiques du voisinage de p_C et μ_G et θ_G celles de p_G. On cherche donc à minimiser la différence entre l'écart-type et la moyenne des luminances des deux voisinages. On note que la valeur des facteurs a peu d'influence sur le résultat.

Une fois que l'échantillon minimisant cette valeur est trouvé, on transfert ses canaux α et β en conservant la luminance d'origine du pixel gris. Après que l'opération soit appliquée sur l'ensemble des pixels de l'image cible, on obtient le résultat final en transformant l'image dans l'espace de couleur RGB.

2.3 Transfert de couleur avec guide

Il est généralement préférable de réduire le nombre d'interventions de l'utilisateur dans le procédé afin d'automatiser plus facilement la colorisation d'un grand nombre d'image. Dans le procédé de base (section 2.2), l'utilisateur fourni une image en niveau de gris à coloriser, ainsi qu'une image de référence en couleur contenant la palette de couleur à transférer. Idéalement, les deux images devraient avoir des textures similaires pour le transfert de couleur. En revanche, il arrive parfois que plusieurs textures soient trop similaires et entraînent des erreurs dans la phase correspondance. Pour cela, les auteurs proposent une version altérée de la méthode où l'utilisateur peut faire correspondre différentes zones des deux images. Les couleurs sont ensuite transférées zone par zone, puis propagées sur le reste de l'image. Le but est donc de définir des zones de correspondances entre les deux images, afin d'aider l'algorithme pendant la phase de transfert sur des images où les textures sont trop similaires. Cela permet aussi de laisser un choix artistique à l'utilisateur s'il souhaite transférer les couleurs d'une texture à une autre, et ce même s'il elles ne sont pas forcément similaire.

Comme dans le procédé général, on commence d'abord par transposer les deux images dans l'espace de couleur $l\alpha\beta$ puis on applique le *remapping* d'histogramme (voir section 2.2.2). Ensuite, pour chaque zone de correspondance définie par l'utilisateur on applique le transfert de couleur décrit dans la partie 2.2.4 en prenant un nombre fixe d'échantillons. La figure 4c illustre l'état de l'image lors de cette étape du procédé. L'ensemble des couleurs désirées ont été transférées vers un certain nombre de zones de l'image en niveau de gris. Dans la suite du procédé, l'image en couleur n'est plus utilisé.

Les couleurs des zones coloriées de l'image sont ensuite propagées dans le reste de l'image. Pour cela, on échantillonne les nouvelles zones coloriés de l'image, et on compare l'ensemble des pixels non coloriés avec les échantillons. L'échantillon idéal est celui qui minimise la distance L_2 entre les luminances du voisinage. Lorsqu'un pixel est au bord de l'image, la taille de la fenêtre de voisinage est redimensionnée pour les deux pixels. Pour deux voisinages V_G et V_C de dimension identique, l'erreur de différence à minimiser est définie ainsi :

$$E(V_1, V_2) = \sum_{n \in V} [p_C - p_G]^2$$

où p_C et p_G sont respectivement les pixels en couleur et en niveau de gris des voisinages V_C et V_G. Une fois l'échantillon idéal trouvé, sa couleur est transféré au pixel non colorié. Comme dans le procédé général, quand l'ensemble des pixels sont coloriés, l'image est retransposée dans l'espace de couleur RGB pour obtenir le résultat final.

3 Résultats expérimentaux

3.1 Implémentation

L'implémentation de l'algorithme est faite en C++ avec la librairie OpenCV. Pour le transfert de couleur avec guide, l'utilisateur trace les zones rectangulaires à la souris directement dans le programme.

3.2 Protocole

La mesure d'un résultat de colorisation est principalement basée sur la perception de l'utilisateur. Dans certains cas il est possible d'obtenir une mesure plus ou moins objective sur la qualité d'un résultat. On peut par exemple, transformer une image de référence vers une image en niveau de gris, puis la coloriser avec l'image de référence. Dans notre implémentation, nous prenons respectivement 30%, 59% et 11% des canaux RGB pour convertir l'image en niveau de gris. Le principe est alors de comparer le résultat avec l'image originale en couleur. Nous utilisons comme métrique le PSNR. Le PSNR est basé sur le logarithme de l'erreur quadratique moyenne des canaux des images. Il est exprimé en décibel. Cette métrique est généralement utilisée pour mesurer la qualité d'algorithmes de compression d'image. Le PSNR est souvent pour ne pas être représentative de la perception humaine [13] mais elle est quand même efficace pour détecter des variations de qualité comme le bruit.

Pour aller plus loin, nous avons aussi envisagé d'utiliser d'autres métriques comme la *Structural Similarity* (SSIM), qui comme le PSNR est utilisée dans le domaine de la compression d'image. Cette métrique mesure les changements dans la structure de l'image en analysant différentes sections de l'image. En partant du principe que la perception humaine est plus sensible aux changements de structure [14], cette métrique est plus représentative de la perception humaine que le PSNR. D'autres métriques comme *LPIPS* [15] évaluent la *similarité perceptuelle* sur plusieurs sections de l'image à l'aide de réseaux de neurones. En revanche, nous n'avons pas eu le temps d'utiliser d'autres métriques que le PSNR.

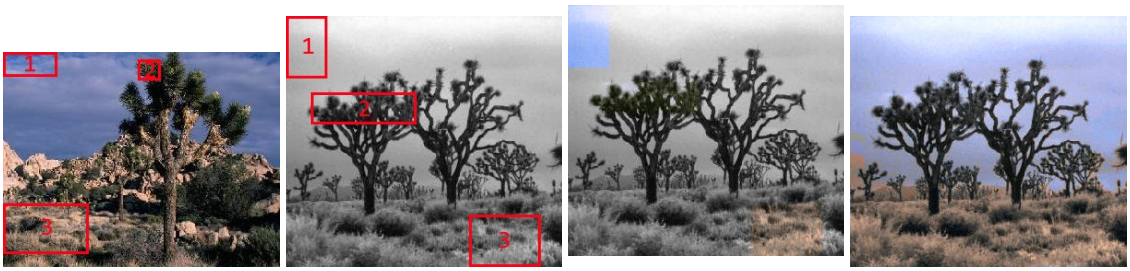
Les bases de données d'images utilisées pour les tests sont Caltech101, Set5 et Set14 ainsi que des images satellites du projet GeoCover de la NASA.

(a) Image de référence (b) Image à coloriser (c) Résultat final

Fig. 2: Exemple du procédé sans guide.

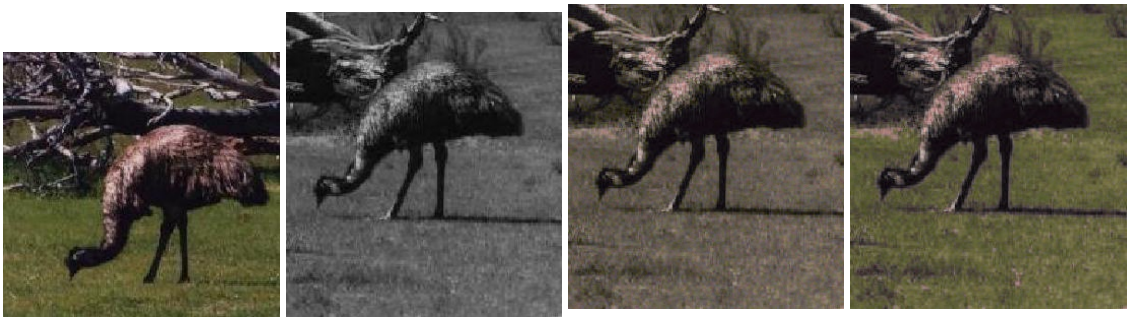
(a) Image de référence (b) Image à coloriser (c) Résultat final

Fig. 3: Exemple du procédé sans guide où l'image à coloriser est obtenue à partir de l'image de référence (voir 3.2).



(a) Image de référence (b) Image à coloriser (c) Étape intermédiaire (d) Résultat final

Fig. 4: Exemple du procédé avec guide. Les images (a) et (b) sont données en entrée avec 3 régions de correspondance. L'image (c) montre l'état de l'image entre la colorisation des régions et la propagation des couleurs.



(a) Image de référence (b) Image à coloriser (c) PSNR maximal (d) PSNR minimal

Fig. 5: Exemple de deux résultats (c) et (d) qui illustrent deux cas extrêmes de PSNR sur 100 essais. Ici 31dB pour (c) et 33dB pour (d)

3.3 Résultats

Nous présentons ici quelques résultats obtenus avec notre implémentation ainsi que ses limites.
La figure 2 montre le résultat de la méthode générale sans guide. On observe que les détails assez fins de la zone urbaine sont difficilement distinguables. Nous avons trouvé que la visibilité de cette zone urbaine dépend principalement de l'échantillonnage de l'image de référence. Les paramètres comme la taille de la fenêtre de voisinage et les poids sur l'écart-type et la moyenne ont une influence minimale sur la perception et le PSNR. Échantillonner toute l'image ne procure pas forcément de meilleurs résultats, mais seulement dans le cas où l'image à coloriser est obtenue à partir de l'image de référence comme décrit dans le protocole (section 3.2). La figure 3 montre un résultat obtenu en suivant ce processus avec 200 échantillons dans l'image de référence.

La figure 4 montre le fonctionnement du procédé avec guide. L'utilisateur fournit 3 zones de correspondances. La figure 4c représente l'état de l'image avant la propagation des couleurs des régions coloriées. On observe des zones où les couleurs se sont mal propagées vers les bords de l'image, dû à la réduction de la taille des fenêtres et la distribution de luminance presque homogène de la texture du ciel.

La figure 5 illustre les effets de la variation d'échantillonage évoqué plus haut. Deux images différentes sont fournies, sans guide, avec comme paramètre 200 échantillons sur l'image de référence. Sur 100 essais sur ces entrées, nous obtenons une variation de PSNR d'environ 2dB, centrée sur 32dB. Les figures 5c et 5d représentent les cas extrêmes. On voit que le cas de PSNR minimal produit une image quasiment monochrome. Ces images sont particulièrement susceptibles aux erreurs de correspondance car les textures d'herbe et du plumage de l'émeu sont très similaires. Un ensemble d'échantillons mal placé peut entrainer une perte de qualité de 2dB.
Avec la méthode générale, les scores PSNR des résultats sont comparables à ceux d'un algorithme de compression avec pertes.

4 Conclusion

Nous avons implémenté la méthode de colorisation de transfert de couleur proposée par Welsh et al. La version de base permet de coloriser une image en fournissant une image en couleur de référence dont la palette de couleur va être transférée à l'image en niveau de gris. La méthode se veut générale, il n'y a donc aucun prérequis sur les images et pas de notion de spatialité. L'utilisateur a aussi la possibilité de guider la colorisation en faisant correspondre des régions des images. Les couleurs sont alors transférées à l'intérieur des régions, puis propagées sur le reste de l'image en utilisant la métrique L_2. Les résultats obtenus ont clairement montré les limites de la méthode, notamment de la propagation des couleurs. Des pistes d'améliorations seraient d'utiliser d'autres métriques pour la propagation de couleur, l'ajout de la notion de spatialité ou bien la segmentation de l'image.

References

[1] Welsh et al. "Transferring colors to grayscale images". In: *ACM Transactions on Graphics* (2002).

[2] K. Pratt. *Digital Image Processing*. John Wiley & Sons, 1991.

[3] Erik Reinhard et al. "Color transfer between images". In: *IEEE Computer Graphics and Applications* (2001), pp. 34–41.

[4] Anat Levin, Dani Lischinski, and Yair Weiss. *Colorization using Optimization*. Tech. rep. 2004.

[5] M. Richart, J. Visca, and J. Baliosian. "Image Colorization with Neural Networks". In: *2017 Workshop of Computer Vision (WVC)*. 2017, pp. 55–60.

[6] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. "Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification". In: *ACM Transactions on Graphics*. 2016.

[7] Jason Antic. *DeOldify: Colorizing and Restoring Photos and Video*.

[8] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. *Image Style Transfer Using Convolutional Neural Networks*. Tech. rep.

[9] Daniel L Ruderman, Thomas W Cronin, and Chuan-Chin Chiao. "Statistics of cone responses to natural images: implications for visual coding". In: *J. Opt. Soc. Am. A* (1998), pp. 2036–2045.

[10] Buckley Robert R and Edward J Giorgianni. "CIELAB for Color Image Encoding". In: *Encyclopedia of Color Science and Technology*. 2016, pp. 213–221.

[11] Le Rohellec Jean Viénot Françoise. "Colorimetry and Physiology - The LMS Specification". In: *Digital color. Acquisition, Perception, Coding and Rendering*. 2013.

[12] Aaron Hertzmann et al. "Image Analogies". In: *SIGGRAPH* (2001).

[13] O. Nemethova et al. "PSNR-based estimation of subjective time-variant video quality for mobiles". In: *MESAQUIN* (2006).

[14] A. Horé and D. Ziou. "Image Quality Metrics: PSNR vs. SSIM". In: *2010 20th International Conference on Pattern Recognition*. 2010, pp. 2366–2369.

[15] Richard Zhang et al. "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric". In: *CVPR*. 2018.