

Projet de traitement d'image

Colorisation d'image



Source : [1]

SOMMAIRE

Introduction	3
I/Etat de l'art	4
1/Transferring Color to Greyscale Images	4
2/Matting et colorisation	5
3/Scribbling	7
4/Automatisation	8
II/ Algorithme implémenté	9
1/Schéma	9
2/Résultats	11
3/Commentaires	13
Conclusion	14
Bibliographie	15

Introduction

Si aujourd’hui l’utilisation du noir et blanc est un procédé artistique, il s’agissait d’un non-choix jusqu’à l’invention des photographies en couleur. La colorisation des images noir et blanc, dans un but de réalisme, est une pratique ancienne, autrefois réalisée en appliquant peinture, aquarelle, ou encore huile teintée à la surface de l’image.

On s’intéressera ici à une approche informatisée du problème, offrant un gain de temps non négligeable. La grande difficulté d’une telle approche est de retrouver une information manquante. En effet, une image en niveau de gris ne contient aucune information de couleur : un pixel en niveau de gris correspondant au ciel par exemple, peut très avoir la même valeur qu’un pixel en niveau de gris correspondant à de l’herbe. Il s’agit donc de trouver une ou des approches permettant une colorisation optimale d’une image en niveau de gris.

Plusieurs méthodes ont été mises au point : certaines utilisent une colorisation dite par « l’exemple », soit un transfert de couleurs depuis une image source donnée, selon différentes approches, d’autres par application par l’utilisateur de couleurs à certains endroits de l’image.

Ces différentes méthodes requièrent une intervention humaine, et on évoquera l’utilité majeure du Machine Learning afin de se passer de cette dernière.

Une implémentation a été réalisée dans le cadre de ce projet, reprenant une des méthodes présentées. L’algorithme sera expliqué, et on discutera des résultats obtenus.

I/Etat de l'art

1/Transferring Color to Greyscale Images

En 2002 paraît le papier de recherche *Transferring Color to Greyscale Images*, de Tomihisa Welsh, Michael Ashikhmin, Klaus Mueller [1]. Cette méthode est souvent considérée comme la méthode pionnière de la colorisation.

Il s'agit d'une colorisation par transfert de couleur: on demande à l'utilisateur de choisir une image à coloriser, ainsi qu'une image « source » en couleur.



Exemple d'une image à coloriser (cible)



Image source

Le système de couleur L*a*b* est le système de couleur qui reproduit le plus fidèlement la perception humaine des couleurs [2]. Il s'agit d'une superposition de 3 matrices de la taille de l'image (comme pour une image RGB), avec en première matrice les valeurs de luminance (L). Les deux autres matrices, contenant respectivement les valeurs des paramètres a et b, expriment l'écart de la couleur par rapport à celle d'une surface grise de même clarté. Les images sont dans un premier temps convertie dans ce système.

Les deux images n'ont pas nécessairement la même luminosité, on souhaitera donc dans un premier temps effectuer une harmonisation de la luminance (« luminance remapping »). Le papier de recherche *Image Analogies*, d'Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, David H. Salesin [3] présente une formule de ce remapping comme suit :

$$Y(p) \leftarrow \frac{\sigma_B}{\sigma_A} (Y(p) - \mu_A) + \mu_B$$

Avec $Y(p)$ la luminance d'un pixel, μ_A et μ_B les moyennes des luminances des images A et B, et σ_A et σ_B les écarts types des luminances.

Ce remapping est appliqué à chaque image par rapport à l'image cible.

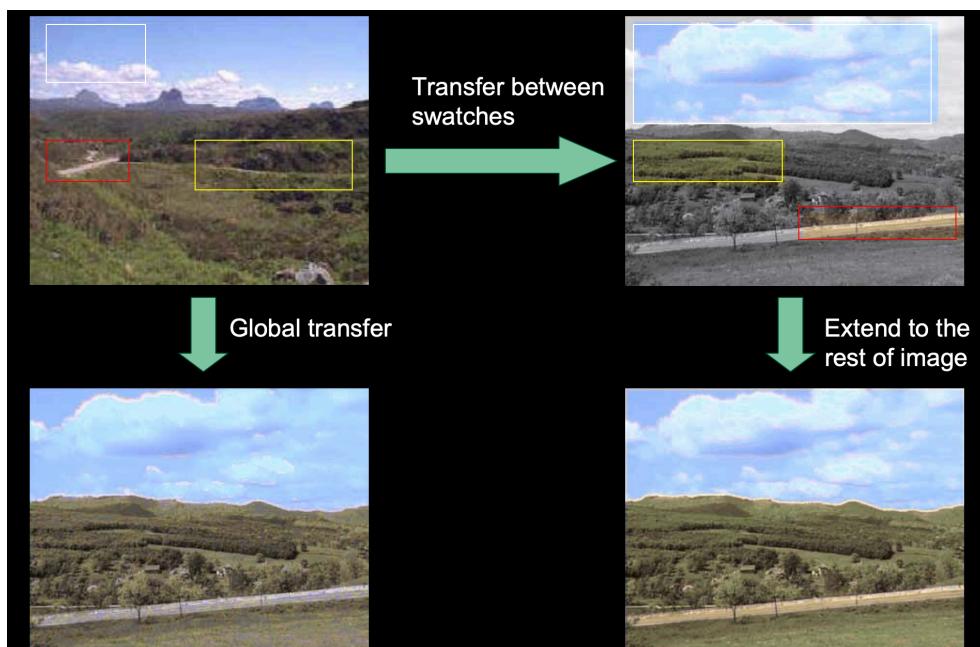
Une seconde information est ensuite associée à chaque pixel : une information de texture [9], donnée par l'écart-type des pixels en niveau de gris au voisinage du pixel analysé.

Il s'agit ensuite de choisir quelle couleur appliquer à chaque pixel. Deux informations sont donc à disposition : la luminance et la texture.

On applique un poids de 50% à chacune de ces informations, puis on recherche le pixel de l'image source qui s'approche le plus du pixel à coloriser selon ces deux critères.

Une première limitation de cette méthode est due au fait qu'entre deux images, les objets ayant la même texture et luminance n'ont pas nécessairement la même couleur. Cette différence peut entraîner une colorisation insatisfaisante, malgré un choix cohérent d'image source.

Une façon de traiter le problème est d'implémenter une méthode complémentaire de « swatch » : en plus de choisir une image source, on demande à l'utilisateur de choisir des zones des deux images dont les couleurs correspondent. On colorise donc les zones de l'image cible avec les couleurs des zones de l'image source associées. Ensuite, le reste de l'image cible est colorisé en utilisant comme source les zones déjà colorisées de l'image cible. En effet, il paraît cohérent de supposer que les correspondances de couleurs entre deux textures et luminances similaires sont plus probables au sein de la même image.

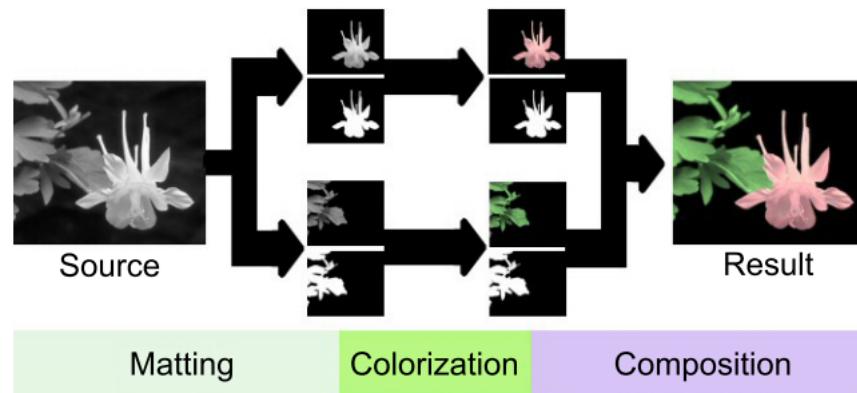


Exemple de colorisation à l'aide swatches [13] : en haut à gauche, l'image source, en bas à droite le résultat d'une colorisation sans swatch. On remarque que la route n'est pas de la couleur attendue.

En haut à droite, on peut voir l'image cible (à coloriser), avec les swatches colorisés depuis ceux correspondant dans l'image source. Enfin, l'image en bas à droite est le résultat de la colorisation du reste de l'image suite à la colorisation des swatches.

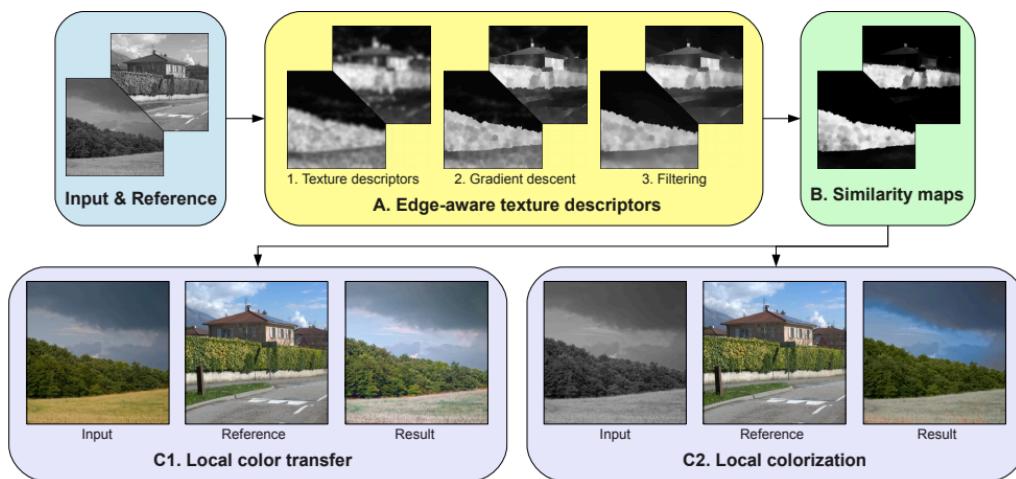
2/Matting et colorisation

Le papier de recherche *Grayscale Image Matting and Colorization*, de Tongbo Chen, Yan Wang, Volker Schillings, Christoph Meinel [4], décrit une méthode permettant un matting (détermination et extraction des différents objets présents sur une image) d'image en niveaux de gris. Dans une visée de colorisation, cette information permet en effet de coloriser objet par objet l'image cible (chaque objet ayant généralement une même couleur dominante). L'idée générale est donc d'utiliser la méthode de colorisation de Welsh (voir I/1/), mais en utilisant cette approche objet par objet.



Exemple : Processus de traitement d'une image de fleur [4]

Dans la même idée générale, *Color Transfer and Colorization based on Textural Properties*, de Benoit Arbelot, Romain Vergne, Thomas Hurtut, Joëlle Thollot [5] permet une colorisation d'image utilisant un transfert de couleur se basant sur les informations de texture de chaque objet extrait de l'image.



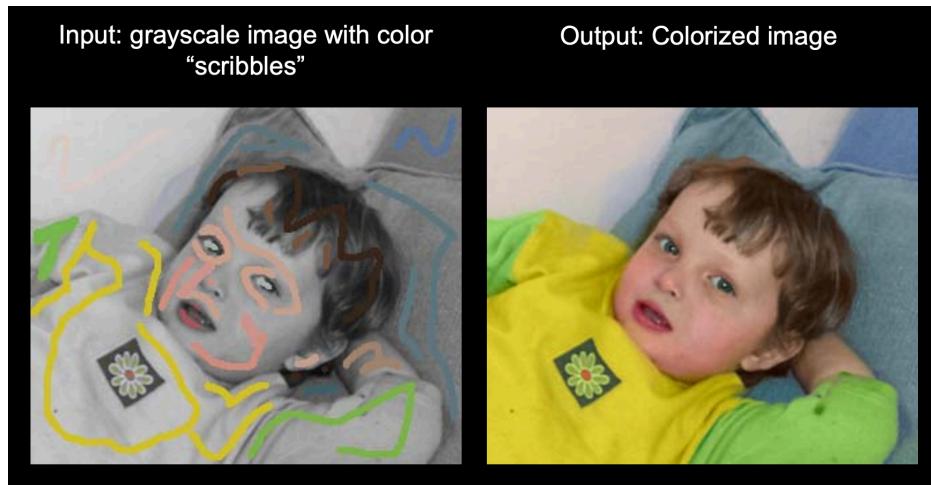
Processus général de traitement d'une image et colorisation [5]

Ces deux méthodes nécessitent toujours une intervention humaine, par exemple la sélection d'une image de référence. Elles offrent cependant une méthode de colorisation plus robuste qu'une méthode utilisant extrayant l'information de texte à partir d'une simple mesure de l'écart-type du voisinage. En effet, une délimitation d'objet permet de déterminer la texture de l'objet dans son intégralité.

Enfin, on peut faire l'analogie entre cette approche par objet et la méthode par swatch, où chaque objet serait un swatch dont le swatch correspondant dans l'image source serait déterminé automatiquement par critère de texture.

3/Scribbling

La méthode décrite dans le papier *Colorization Using Optimization*, de Anat Levin, Dani Lischinski, Yair Weiss [6], contrairement à celles présentées précédemment, n'utilise pas une colorisation à partir d'une image source mais à partir de « scribble » (serpentins de couleurs).



Exemple : gauche : image à coloriser, avec des « scribbles » de couleur appliqués, droite : résultat de la colorisation

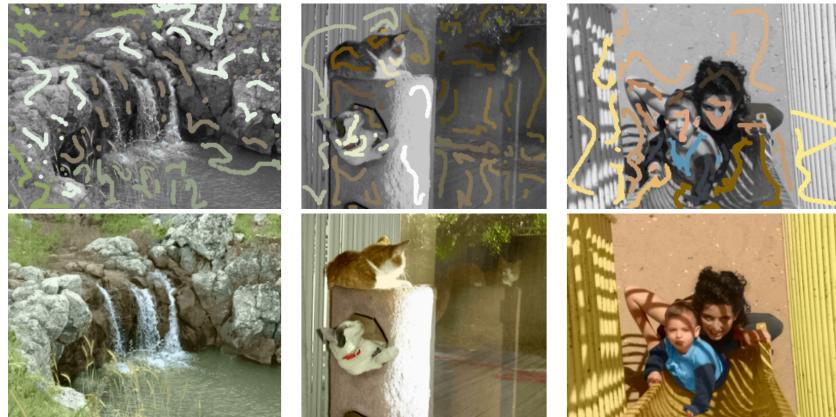
Cette méthode repose tout d'abord sur l'application par l'utilisateur de couleurs par zones.

Le raisonnement derrière la colorisation totale de l'image est le suivant : deux pixels voisins d'intensité lumineuse similaire devrait avoir des couleurs similaires. Ce principe est formalisé mathématiquement sous la forme suivante :

$$J(U) = \sum_{\mathbf{r}} \left(U(\mathbf{r}) - \sum_{\mathbf{s} \in N(\mathbf{r})} w_{rs} U(\mathbf{s}) \right)^2$$

Avec U la chrominance (système de couleur YUV), r le pixel à analyser, $N(r)$ l'entourage du pixel r , w_{rs} la fonction de poids (grande quand $Y(r)$ et $Y(s)$ sont similaires, petite sinon) et $U(s)$ la chrominance du pixel s .

On obtient les résultats suivants :



Exemples de résultats de colorisation

Quand bien même une faible information de scribble peut suffire à coloriser une image dans un bon nombre de cas, cette méthode reste cependant très contraignante et n'est pas automatisable.

4/Automatisation

Toutes les méthodes précédentes présentent un inconvénient majeur : elles requièrent une participation active de l'utilisateur.

Grâce aux avancées en *Machine Learning*, cette étape peut maintenant être automatisée à l'aide de réseaux de neurones.

Un exemple d'automatisation est présenté dans le papier *Colorful Image Colorization*, de Richard Zhang, Phillip Isola, Alexei A. Efros [7]. Il s'agit d'une méthode ne demandant plus aucune intervention humaine, et qui permet de coloriser automatiquement des images en niveaux de gris en image aux couleurs plausibles.



Exemples d'images colorisées grâce à cette méthode, sans image de référence (cohérente avec l'image à coloriser) sélectionnée par l'utilisateur.

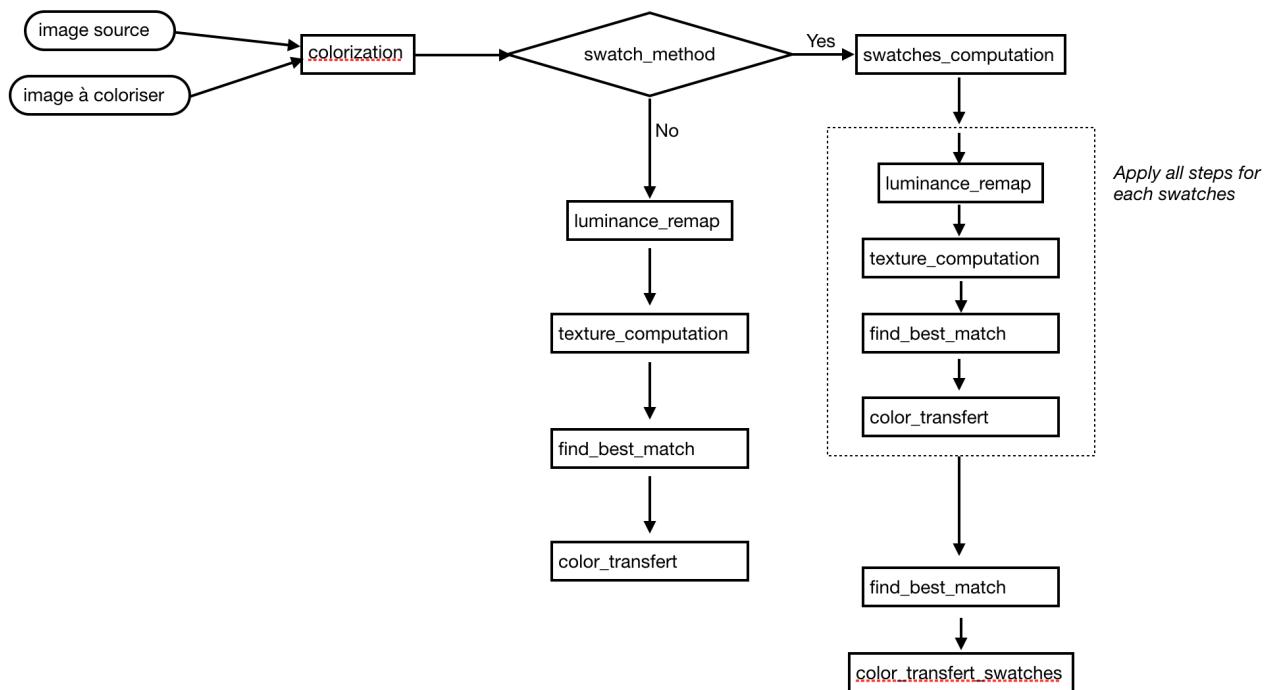
On peut aussi citer le site *Colourise* [8] qui permet, en quelques secondes et en se basant sur du *Deep Learning*, de coloriser des images en niveaux de gris.

II/Algorithme implémenté

On choisit ici d'implémenter la méthode *Transferring Color to Greyscale Images*, de Tomihisa Welsh, Michael Ashikhmin, Klaus Mueller présentée en partie I/ 1/. Comme expliqué précédemment, cette approche est considérée comme la base de la colorisation des images, et sert souvent de fondation ou de comparaison pour les autres méthodes. Elle présente aussi l'avantage d'être relativement simple à appréhender, et permet de rendre compte efficacement des problématiques liées à la colorisation d'une image.

L'implémentation a été faite sur Matlab 2018b.

1/Schéma



La fonction principale est **colorization.m**, et prend comme argument le nom de l'image source ainsi que le nom de l'image à coloriser, qui doivent donc se trouver dans le répertoire courant de Matlab (ex : **colorization('tree_c.jpg','forest_g.jpg')**).

La fonction a deux façons d'effectuer la colorisation : avec ou sans les swatches. Le choix de la méthode sera proposé en ligne de commande.

a/ Sans swatch method

Dans un premier temps, l'image source sera convertie au format L*a*b dans la variable **csource_lab**. Elle sera aussi convertie en noir et blanc dans la variable **gsource_lab**. L'image à coloriser sera stockée dans la variable **gtarget**.

La première étape est la fonction **luminance_remap** : cette dernière permet l'harmonisation des luminances, toutes rapportées à celle de l'image à colorer.

Dans un second temps, la fonction **texture_computation** permet d'obtenir une information dite de « texture » de chaque pixel (donnée par l'écart-type des valeurs des

pixels alentours, valeur de voisinage ici fixée à 5 comme explicitée dans le papier de recherche sur lequel se base l'implémentation [1]).

A partir de ces deux informations, la fonction **find_best_match** cherchera, pour chaque pixel de l'image à coloriser, le pixel qui lui correspond le plus dans **gsource_lab**, en comparant les valeurs de luminance et de texture de chacun (chaque information ayant un poids de 1/2 [1]). Cette fonction a comme sortie un vecteur **best_match** contenant, dans l'ordre des pixels de l'image à coloriser, les indices des pixels analogues de l'image source.

Enfin, la fonction **color_transfert** permet de transférer les valeurs des pixels analogues de l'image source à l'image cible.

b/ Avec swatch_method

La fonction **swatch_computation** permet à l'utilisateur de sélectionner les swatches concordant sur l'image source puis sur l'image à coloriser. Pour ce faire, on demandera d'abord à l'utilisateur via la fenêtre de commande le nombre de swatches à considérer. Ensuite, on affiche les images source et cible dans une même fenêtre, et l'utilisateur doit sélectionner en traçant un rectangle, d'abord sur l'image source, un premier swatch, puis sur l'image cible, le swatch correspondant au premier (et ainsi de suite jusqu'à ce que le nombre de swatch voulu soit atteint).

Dans un second temps, on colorise chaque swatch de l'image cible avec comme source, le swatch de l'image source correspondant : nous sommes ainsi sur que les couleurs seront les bonnes.

Une fois tous les swatch colorisés, on concatène toutes leurs valeurs de luminance et de texture de ces derniers dans les variables **ctarget_luminance** et **ctarget_texture**, puis on appelle **find_best_match** avec comme argument ces dernières, ainsi que les valeurs de luminance **gtarget_luminance** et de texture **gtarget_texture** de l'image à coloriser. Cette fonction aura comme sortie un vecteur **best_match** contenant, dans l'ordre des pixels de l'image à coloriser, les indices des pixels analogues des swatches déjà colorisés.

Color_transfert_swatches permet le transfert correct des couleurs depuis les swatches colorés précédemment.

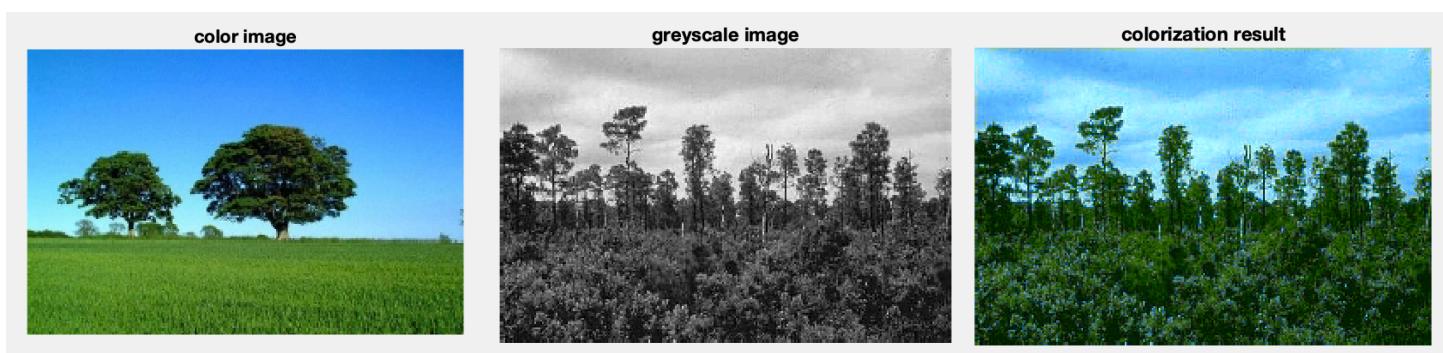
2/Résultats



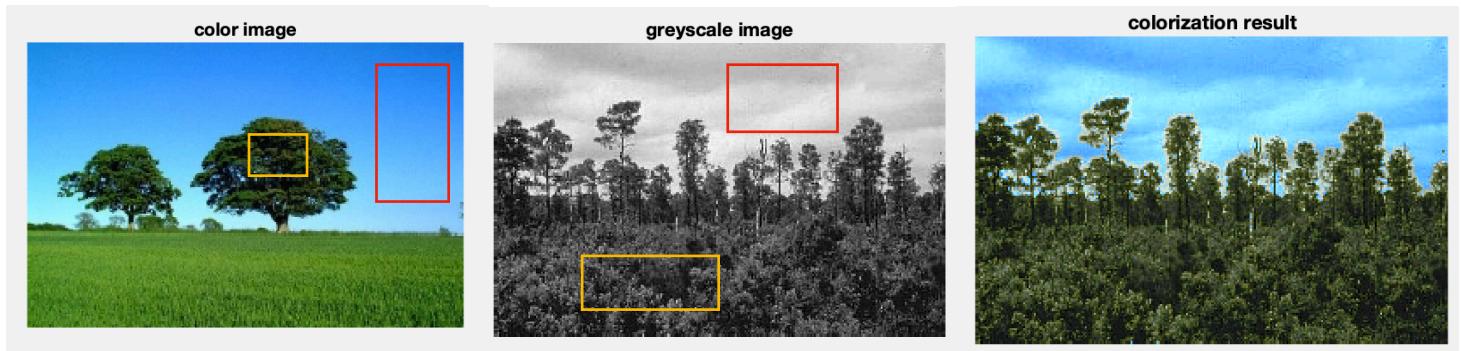
Résultat 1 : Exemple de colorisation réussie, sans swatch (*girl_c.jpeg*, *girl_g.jpeg*)



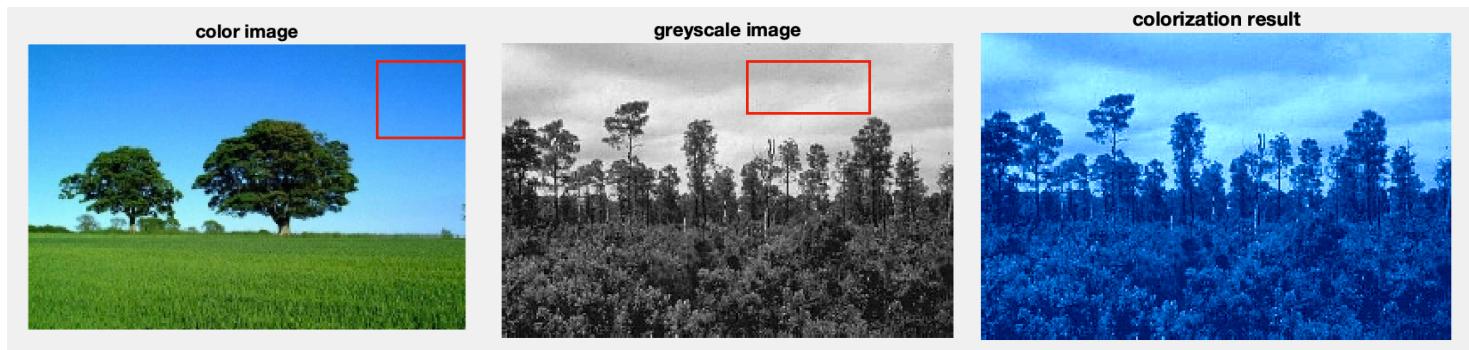
Résultat 2 : Exemple de colorisation défectueuse, sans swatch (*desert_c.jpeg*, *desert_g.jpeg*)



Résultat 3 : Exemple de colorisation partiellement réussie, sans swatch (*tree_c.jpeg*, *forest_g.jpeg*)



Résultat 4 : Exemple de colorisation réussie, avec swatches (*tree_c.jpeg*, *forest_g.jpeg*)



Résultat 5 : Exemple de colorisation défectueuse, avec un swatch (*tree_c.jpeg*, *forest_g.jpeg*)

3/Commentaires

On remarque pour le Résultat 2 la présence d'orange dans le ciel et de bleu dans les dunes, attestant bien qu'avec seulement les informations de texture par voisinage direct et de luminance, on ne peut parfois pas obtenir de résultat satisfaisant. Ici, par simple comparaison de luminance et de texture entre les pixels en haut à gauche de l'image à coloriser et les pixels composant l'image source, on trouve que la meilleure correspondance est obtenue pour des pixels qui correspondaient sur l'image source à du sable.

Le Résultat 3 paraît satisfaisant à première vue, mais on peut remarquer la présence de taches bleues dans la forêt. Le Résultat 4, colorisant la même image mais cette fois en utilisant la méthode des swatches permet de corriger efficacement ce problème.

Le Résultat 5 permet de mettre en lumière un problème majeur de l'utilisation des swatches : la colorisation finale de l'image ne se basant que sur la colorisation de ces derniers, ils contiennent les seules couleurs qui viendront coloriser l'image. Dans cet exemple, on a choisi un seul swatch qui contenait du bleu, toute l'image à coloriser est donc dans des tons bleus.

Dans le cadre d'une image contenant peu de couleurs différentes, peu de swatches suffisent donc, mais pour une image plus « complexe », contenant une palette de couleurs plus riches, il faudra à l'utilisateur sélectionner un nombre conséquent de swatches si l'on souhaite une colorisation correcte de l'image. La méthode des swatches est donc déconseillée pour les images très colorées.

D'une façon plus générale, cette méthode requiert une participation humaine active, que ce soit par la sélection d'une image source cohérente avec l'image que l'on souhaite coloriser, ou, le cas échéant, par la sélection de différents swatches.

Une approche automatique permettrait de se passer de ces interventions, comme évoqué plus haut.

Conclusion

La problématique de la colorisation des images est ancienne. Les premières applications visées étaient le passage en couleurs de photographies et d'images d'archives initialement en noir et blanc.

Les méthodes initiales sans utilisation de Deep Learning, et en particulier les premières méthodes référencées ici, requièrent une participation active de l'utilisateur.

L'identification des textures, la décomposition de l'image par objet, le transfert de couleur depuis une image source cohérente permettent une implication réduite de l'utilisateur en comparaison à la l'application directe de peinture sur une image noir et blanc. L'intervention humaine reste néanmoins toujours nécessaire.

La méthode implémentée permet d'obtenir des résultats satisfaisants, plus explicitement une colorisation plausible, à condition cependant de fournir une image source cohérente avec l'image à coloriser. Une telle condition demeure une forte contrainte et nécessite une intervention humaine non négligeable.

Bibliographie

[1] Transferring Color to Greyscale Images, de Tomohisa Welsh, Michael Ashikhmin, Klaus Mueller, 2002

<https://webserver2.tecgraf.puc-rio.br/~scuri/inf1378/pub/welsh.pdf>

[2] L*a*b* CIE 1976, article Wikipédia

https://fr.wikipedia.org/wiki/L*_a*_b*_CIE_1976

[3] Image analogies, Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, David H. Salesin, 2001

<https://mrl.nyu.edu/publications/image-analogies/analogies-72dpi.pdf>

[4] Grayscale Image Matting and Colorization, de Tongbo Chen, Yan Wang, Volker Schillings, Christoph Meinel, 2004

<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1.4710&rep=rep1&type=pdf>

[5] Color Transfer and Colorization based on Textural Properties, de Benoit Arbelot, Romain Vergne, Thomas Hurtut, Joëlle Thollot, 2016

<https://hal.archives-ouvertes.fr/hal-01246615/file/RR-8834.pdf>

[6] Colorization Using Optimization, de Anat Levin, Dani Lischinski, Yair Weiss, 2004

<http://webee.technion.ac.il/people/anat.levin/papers/colorization-siggraph04.pdf>

[7] Colorful Image Colorization, de Richard Zhang, Phillip Isola, Alexei A. Efros, 2016

<https://arxiv.org/pdf/1603.08511.pdf>

[8] Colourise, site permettant la colorisation d'image en niveau de gris

<https://colourise.sg/>

[9] Digital Image Processing Course Book

http://130.243.105.49/Research/Learning/courses/dip/2011/lectures/DIP_2011_L14.pdf

Non cité explicitement :

[10] Colorization of grayscale images and videos using a semiautomatic approach, de Vivek George Jacob and Sumana Gupta

<https://projet.liris.cnrs.fr/imagine/pub/proceedings/ICIP-2009/pdfs/0001653.pdf>

[11] Creating the Color Panoramic View using Medley of Grayscale and Color Partial Images, de Dr. H. B. Kekre, Sudeep D. Thepade

<https://pdfs.semanticscholar.org/242c/38352ce403be029caa8a0be82260a406e43d.pdf>

[12] Automatic Image Colorization via Multimodal Predictions, de Guillaume Charpiat, Matthias Hofmann, Bernhard Schölkopf

<https://www.lri.fr/~gcharpia/colorisation.pdf>

[13] Signal Processing Course, Department of Computer Science, University of North Carolina at Chapel Hill

https://www.cs.unc.edu/~lazebnik/research/fall08/lec06_colorization.pdf