Transfert de couleurs et colorisation guidés par la texture

Benoit Arbelot, Romain Vergne, Thomas Hurtut, Joëlle Thollot

Transfert de couleurs et colorisation guidés par la texture

B. Arbelot[1], R. Vergne[1], T. Hurtut[2] and J. Thollot[1]

[1]Inria-LJK (UGA, CNRS), France
[2]Polytechnique Montréal, Canada

Transfert de couleurs Colorisation

Figure 1 : *Notre algorithme permet d'appliquer automatiquement un transfert de couleurs local (à gauche) ou une colorisation (à droite) en se basant sur les propriétés texturelles des images.*

Résumé

Cet article se concentre sur deux problèmes de manipulation de couleurs liés : le transfert de couleurs *qui modifie les couleurs d'une image, et la* colorisation *qui ajoute des couleurs à une image en niveaux de gris. Les méthodes automatiques pour ces deux applications modifient l'image d'entrée à l'aide d'une image de référence contenant les couleurs désirées. Les approches précédentes visent rarement les deux problèmes simultanement et souffrent de deux principales limitations : les correspondances créées entre les images d'entrée et de référence sont incorrectes ou approximatives, et une mauvaise cohérence spatiale autour des structures de l'image. Dans cet article, nous proposons un pipeline unifiant les deux problèmes, basé sur le contenu texturel des images pour guider le transfert ou la colorisation. Notre méthode introduit un descripteur de textures préservant les contours de l'image, basé sur des matrices de covariance, permettant d'appliquer des transformations de couleurs locales. Nous montrons que notre approche est capable de produire des résultats comparables ou meilleurs que d'autres méthodes de l'état de l'art dans les deux applications.*

Abstract

This paper targets two related color manipulation problems : Color transfer *for modifying an image colors and* colorization *for adding colors to a greyscale image. Automatic methods for these two applications propose to modify the input image using a reference that contains the desired colors. Previous approaches usually do not target both applications and suffer from two main limitations : possible misleading associations between input and reference regions and poor spatial coherence around image structures. In this paper, we propose a unified framework that uses the textural content of the images to guide the color transfer and colorization. Our method introduces an edge-aware texture descriptor based on region covariance, allowing for local color transformations. We show that our approach is able to produce results comparable or better than state-of-the-art methods in both applications.*

Mots clé: Transfert de couleurs, Colorisation

1. Introduction

Dans cet article, nous proposons une méthode pour coloriser ou modifier les couleurs d'une image automatiquement. Coloriser une image en noir et blanc, ou modifier les couleurs d'une image à la main pour obtenir une certaine atmosphère est complexe, long et demande une certaine expertise. Les méthodes par l'exemple offrent une alternative intéressante en modifiant automatiquement les couleurs d'une image en **entrée** en fonction d'un exemple (**référence**) contenant les couleurs désirées. La difficulté de ces méthodes est de mettre précisément en correspondance le contenu des images d'entrée et de référence.

Les premiers algorithmes de transfert de couleurs, basés sur des approches globales, transforment l'histogramme de l'image d'entrée pour le rapprocher de celui de l'image de référence. Ces approches sont simples et efficaces avec des images d'entrée bien choisies, cependant elles ont tendance à transférer entre régions parfois mal choisies créant des résultats imprédictibles. De plus, ces approches ne sont pas adaptées au problème de la colorisation où l'image d'entrée n'a pas d'histogramme de couleurs à modifier.

D'autres approches, locales, segmentent les images en différentes sous-régions transférées ou colorisées indépendamment. Ces régions peuvent être définies par l'utilisateur, ou calculées automatiquement à l'aide de descripteurs.

Notre approche est automatique et basée sur des régions définies comme des zones de contenu texturel similaire. Ce choix est dû au fait que les textures sont présentes partout dans la nature, et donc dans beaucoup de photographies. D'autre part, des études perceptuelles ont montré que les premières étapes du système visuel humain sont composées de différents filtres analysant les textures et les variations de couleur dans notre champ de vision [YJ*93, Bal06]. Cela suggère donc que les textures sont importantes lorsque nous regardons des images, ce qui en fait une base pertinente pour appliquer des transformations de couleurs locales. De plus, les textures peuvent être décrites efficacement par un résumé de statistiques de premier et second ordre. Elles présentent donc un entre-deux intéressant entre des descripteurs de bas niveau (comme la luminance ou la chromaticité) qui ne peuvent pas décrire efficacement des zones texturées, et des descripteurs de haut niveau (décrivant des objets et leur sémantique) complexes, pas toujours fiables, et lents à calculer.

Dans le but d'appliquer un transfert entre régions texturées, nos descripteurs sont calculés sur une large échelle afin de pouvoir identifier de larges textures. Cependant, ils doivent aussi respecter et préserver la structure de l'image. Les méthodes existantes pour la décomposition d'une image entre structure et texture ne sont pas idéales pour notre application : les descripteurs préservant les contours (comme le filtre bilatéral) ont du mal à analyser des textures très contrastées et risquent d'introduire des discontinuités intra-textures durant le transfert. L'alternative consistant à détecter les variations d'un descripteur de texture (comme la covariance d'une région) a tendance à flouter les contours de l'image, ce qui crée des halos durant le transfert.

Notre solution pour identifier les textures est fondée sur une analyse texturale, suivie d'un post-traitement préservant les contours afin de créer un descripteur de texture préservant précisément les contours de l'image. Nos contributions peuvent être résumées comme suit :

- Une méthode pour calculer des descripteurs de texture précis, préservant la structure de l'image. Les descripteurs de texture sont calculés à partir de statistiques de premier et second ordre sur la luminance des images, puis des traitements préservant les contours sont appliqués sur ces descripteurs pour préserver la structure de l'image.
- Un pipeline générique pour appliquer un transfert de couleur ou une colorisation fondés sur des propriétés texturales.

2. Travaux précédents

Dans cette section, un résumé des techniques existantes de transfert de couleur et de colorisation est proposé, suivi d'une discussion sur différentes approches pour extraire et analyser les textures.

Transfert de couleur. Le transfert de couleur consiste à modifier les couleurs d'une image d'entrée pour les rapprocher des couleurs d'une image de référence. Cette technique fut initialement introduite par Reinhard *et al.* [RAGS01] comme un simple transfert d'histogramme où la moyenne et la variance de chaque canal de couleur sont transférées séparément, utilisant l'espace de couleur décorrelé $L\alpha\beta$. Cette méthode simple peut être efficace si les images d'entrée sont bien choisies et assez similaires. Une composante de rotation fut ajoutée au transfert par Xiao et Ma [XM06], permettant d'effectuer le transfert dans un espace de couleur corrélé comme RGB. Au lieu de traiter chaque canal indépendamment, Pitié *et al.* [PKD07] proposent de transformer des histogrammes 3D directement en utilisant itérativement des transformations 1D. Bien que la correspondance entre l'histogramme de référence et l'histogramme résultant soit excellente dans cette approche, elle est presque "trop parfaite" et tend à produire des artefacts dans le résultat, le forçant à contenir le même nombre de pixels de chaque

couleur que la référence. Enfin, une approche plus récente basée sur une transformation à plusieurs échelles de l'histogramme a été proposée dans [PR11] où l'utilisateur peut choisir à quel point la correspondance entre les histogrammes doit être précise. De manière générale, ces méthodes globales sont simples mais n'assurent pas que les couleurs soient transférées entre régions similaires. Quand ces méthodes automatiques ne suffisent pas, une segmentation de l'image peut être fournie par l'utilisateur pour transférer uniquement entre régions pré-définies [DX09, AP10, LSZ12].

Dans le but d'effectuer un transfert local automatiquement, Tai *et al.* [TJT05] proposent d'utiliser un mélange Gaussien pour segmenter les images d'entrée et transférer les couleurs entre régions de luminance similaire. Une méthode pour ajuster les couleurs d'une vidéo, basée sur un transfert de couleur entre séquences a été proposée dans [BSPP13]. Leur transformation segmente les images en utilisant la luminance et transfert les couleurs entre les régions d'ombres, de tons moyens, et de surbrillance. Dans une approche similaire, Hristova *et al.* [HLMCB15] divisent les images en nuages Gaussiens se basant sur leur principale caractéristique (luminance ou couleurs). Bien que ces approches soient plus précises que des transferts globaux, elles sont uniquement basées sur des informations de premier ordre pour segmenter les images et ne prennent pas en compte d'information d'ordre plus élevé pour effectuer les correspondances entre les régions des images. En conséquence, des régions ayant des textures différentes mais des luminances similaires ne peuvent pas être différenciées.

Colorisation. La colorisation consiste à ajouter des couleurs (ou de la chrominance) à une image en niveaux de gris. Une des premières approches à attaquer ce problème se base sur des coups de pinceau colorés fournis par l'utilisateur et les étend sur des régions de même luminance à l'aide d'une optimisation [LLW04]. Cette optimisation est utilisée avec des coups de pinceau générés automatiquement dans beaucoup de méthodes de colorisation par l'exemple [ICOL05, GCR*12, KCP15]. Comme elles sont basées sur une optimisation via la luminance dans leur dernière étape, ces méthodes peinent à gérer des textures très contrastées où l'optimisation ne propage pas correctement les couleurs. Plus récemment, Jin *et al.* [JCT14] ont proposé un algorithme aléatoire pour mieux transférer des distributions de couleurs entre régions définies par l'utilisateur. Plus proche de notre approche, d'autres méthodes se basent sur des information d'ordre supérieur pour transférer de la chrominance entre pixels ayant les mêmes statistiques [WAM02, CHS08, BT12]. Cependant, ces méthodes produisent souvent des halos créés par la fenêtre utilisée dans le calcul des statistiques. De plus, ces méthodes sont basées sur une minimisation d'énergie qui les rend typiquement lentes et difficiles à utiliser sur de grandes images.

Analyse de textures. Beaucoup de descripteurs différents ont été utilisés pour manipuler des images en fonction de leur contenu texturel. Des méthodes précédentes de colorisation automatique utilisent des descripteurs SURF, des filtres de Gabor ou des histogrammes de gradients orientés (HoG) comme outils de base pour l'analyse des textures [CHS08, GCR*12, KCP15]. Ces descripteurs sont connus pour être efficaces et discriminants, mais aussi complexes à calculer et à stocker à cause de leur grand nombre d'éléments. De même, les descripteurs de texture basés sur la forme introduits dans [XDG10, XHJF12], bien qu'ils possèdent de nombreuses invariances, sont trop complexes pour une application en manipulation d'image où l'on souhaite obtenir des résultats en un temps raisonnable pour des images de grande taille. Les approches récentes proposées dans [XYXJ12, CLKL14] séparent précisément les textures de la structure mais leur descripteur (*relative total variation*) n'est pas assez précis pour distinguer différentes textures entre elles. Enfin, Karacan *et al.* [KEE13] ont proposé d'utiliser la covariance d'une région comme descripteur de texture pour lisser les images. Notre méthode se base aussi sur une variante de ce descripteur compact et efficace pour décrire les propriétés texturelles. Un de ses désavantages est que ce descripteur à tendance à être peu précis autour des contours de l'image et des transitions entre textures, en particulier quand il est estimé à partir d'un grand voisinage. Pour cette raison, des méthodes de filtrage préservant les contours pouvant être utilisées pour résoudre ce problème sont aussi discutées.

Les filtres préservant les contours sont cruciaux pour respecter les structures d'une image lors d'un lissage, débruitage, d'une accentuation des détails, ou de l'extraction des textures d'une image. Une approche populaire dans ce but est le filtre bilatéral [TM98], qui lisse efficacement les images en préservant les contours en fonction de leur luminance. Cependant, ce filtre a tendance à introduire des halos et des inversions de gradient autour des contours pouvant modifier les informations texturelles. Le filtre guidé (*guided filter*) [HST13] offre une approche différente en utilisant une transformation linéaire d'une image guide pour filtrer une image d'entrée, mais tend aussi à produire des halos autour des contours de l'image. La diffusion anisotrope [PM90] ou le filtre bilatéral non-normalisé [APH*14] sont plus appropriés pour nos descripteurs car ils évitent halos et inversions de gradient quand le lissage est nécessaire à grande échelle.

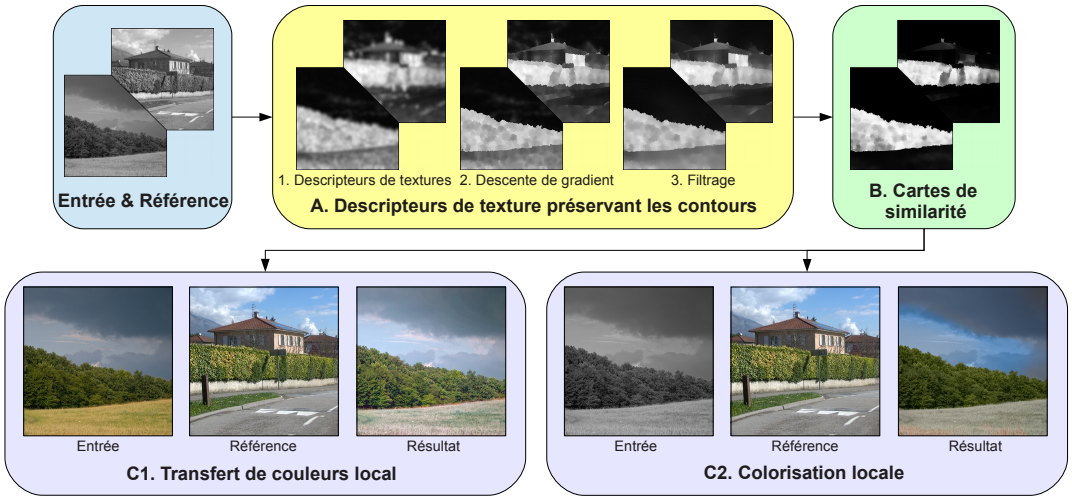


Figure 2 : *Pipeline de notre méthode.* *Des descripteurs préservant les contours sont d'abord calculés pour décrire précisément le contenu texturel des images d'entrée et de référence (A). Ensuite ces descripteurs sont utilisés pour calculer des distances entre pixels et associer les régions similaires, comme montré ici pour la végétation (B). Enfin ces cartes de distance sont utilisées pour le transfert de couleur (C1) et la colorisation (C2), où les couleurs sont attribuées en fonction de la similarité entre pixels.*

3. Notre approche

Notre approche pour éditer automatiquement les couleurs en se basant sur les textures est résumée dans la figure 2. D'abord, des descripteurs sont calculés pour les images d'entrée et de référence en trois étapes (A) : les matrices de covariance de plusieurs caractéristiques locales sont calculées à grande échelle pour définir le contenu texturel de chaque région (A.1). Une descente de gradient multi-échelle déplace ensuite ces descripteurs localement pour retrouver les contours des textures perdus lors de l'analyse à grande échelle (A.2). Enfin, un filtrage préservant les contours est appliqué pour obtenir des descripteurs décrivant précisément des régions de textures homogènes, tout en préservant le détail des transitions entre textures (A.3).

Nos descripteurs permettent de calculer des similarités entre pixels. Grâce à ça, ils permettent de segmenter les images de manière douce, préservant les structures lisses ou nettes. Ceci est illustré dans la figure 2 (B), où la végétation est isolée automatiquement dans les images d'entrée et de référence. Enfin, des cartes de similarité contrôlent le transfert local de couleurs entre les images (C1) ou la colorisation en fonction des régions texturées de manière similaires (C2). Le reste de cet article est organisé comme suit : les descripteurs sont détaillés dans la section 4 et les algorithmes de manipulation de couleurs sont décrits dans la section 5. Des résultats et comparaisons sont présentés en section 6 avant de conclure en section 7.

4. Descripteurs de texture préservant les contours

4.1. Descripteurs de texture locaux

Pour analyser la texture sous-jacente à chaque pixel de l'image d'entrée et de référence, nous avons choisi d'utiliser des matrices de covariances (*region covariance*) [TPM06, KEE13]. En effet, c'est un moyen efficace et compact de décrire les régions d'une image. La covariance d'une région capture la texture sous-jacente en calculant un ensemble réduit de statistiques de second ordre sur des caractéristiques spécifiques de l'image comme la luminance ou le gradient. Considérons un pixel $\mathbf{p}$, décrit par un vecteur $\mathbf{z}(\mathbf{p})$ de caractéristiques de dimension d. La covariance de la région correspondante est une matrice $d \times d$ définie comme suit :

$$\mathbf{C}_r(\mathbf{p}) = \frac{1}{W} \sum_{\mathbf{q} \in N_r^{\mathbf{p}}} (\mathbf{z}(\mathbf{q}) - \boldsymbol{\mu}_r)(\mathbf{z}(\mathbf{q}) - \boldsymbol{\mu}_r)^T w_r(\mathbf{p}, \mathbf{q}),$$

où $N_r^{\mathbf{p}}$ est un voisinage carré centré sur $\mathbf{p}$ de taille $(2r+1) \times (2r+1)$ et $\boldsymbol{\mu}_r$ est un vecteur contenant la moyenne de chaque caractéristique à l'intérieur de cette région. Contrairement à [TPM06], nous ajoutons une fonction de poids Gaussienne d'écart-type $r/3$ assurant que les descripteurs soient définis continûment de pixel en pixel : $w_r(\mathbf{p}, \mathbf{q}) = exp(\frac{-9\|\mathbf{q}-\mathbf{p}\|^2}{2r^2})$. Cette fonction de poids est aussi utilisée pour calculer

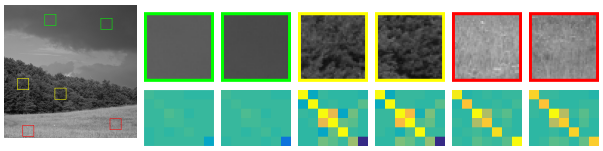
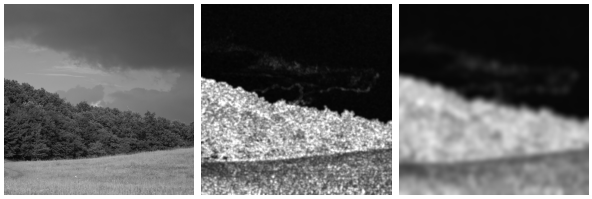


Figure 3 : *Descripteurs de textures.* *Des extraits de régions diverses pris dans l'image de la figure 2 (en haut) et leurs descripteurs correspondants calculés pour le pixel central de la région (en bas). Les extraits issus de régions similaires ont des descripteurs similaires.*



(a) Entrée (b) $\boldsymbol{\mathcal{S}}_6$ (c) $\boldsymbol{\mathcal{S}}_{21}$

Figure 4 : *Descripteurs.* *De petites échelles créent des descripteurs bruités (b). De grandes échelles permettent d'obtenir de descripteurs plus homogènes mais lissent les transitions nettes entre textures. Pour clarifier la visualisation, seul le premier élément de $\boldsymbol{\mathcal{S}}_r$ est représenté (la première valeur de $\mathbf{L}_r^1$) mais le reste de l'ensemble présente le même comportement.*

la moyenne des caractéristiques $\boldsymbol{\mu}_r$. W est un facteur de normalisation : $W = \sum_{\mathbf{q}\in N_r^{\mathbf{p}}} w_r(\mathbf{p},\mathbf{q})$. Nous utilisons $r \in [20,30]$ et utilisons un vecteur 5D basé sur des dérivées de la luminance afin de capturer les textures d'images naturelles à grande échelle :

$$\mathbf{z}(\mathbf{p}) = \left[L(\mathbf{p}) \quad \frac{\partial L(\mathbf{p})}{\partial x} \quad \frac{\partial L(\mathbf{p})}{\partial y} \quad \frac{\partial^2 L(\mathbf{p})}{\partial x^2} \quad \frac{\partial^2 L(\mathbf{p})}{\partial y^2} \quad \frac{\partial^2 L(\mathbf{p})}{\partial x \partial y} \right],$$

où $L(\mathbf{p})$ est la luminance du pixel $\mathbf{p}$. En pratique, chaque caractéristique est d'abord normalisée (divisée par son écart-type) de manière à ce que toutes contribuent de la même manière à l'analyse.

Comme expliqué dans [HCS*09, KEE13], la covariance d'une région ne décrit que des statistiques de deuxième ordre, ce qui peut limiter la nature des textures descriptibles. De plus, calculer des distances entre matrices de covariance est complexe car elles ne sont pas dans un espace Euclidien. Pour ces deux raisons, nous adoptons la solution proposée par Karacan *et al.* [KEE13] qui utilisent une décomposition de Cholesky pour transformer les matrices de covariance en vecteurs pouvant facilement être comparés et enrichis avec des statistiques de premier ordre. Notre descripteur est représenté par :

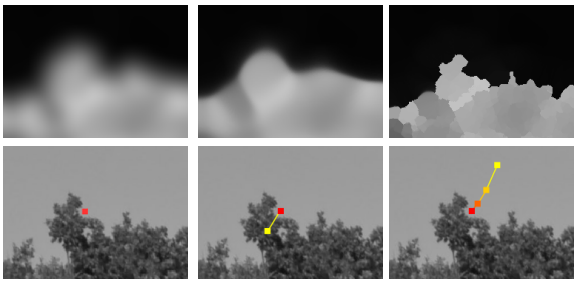
$$\boldsymbol{\mathcal{S}}_r = \begin{pmatrix} \mathbf{L}_r^1 & \cdots & \mathbf{L}_r^d & \boldsymbol{\mu}_r \end{pmatrix}, \tag{1}$$

où $\mathbf{L}_r^i$ est la i^{eme} colonne de la matrice triangulaire inférieure $\mathbf{L}_r$ obtenue par la décomposition de Cholesky $\mathbf{C}_r = \mathbf{L}_r\mathbf{L}_r^T$ à l'échelle r et $\boldsymbol{\mu}_r$ sont les moyennes des caractéristiques dans la région correspondante, c'est-à-dire des statistiques de premier ordre.

Des visualisations de nos descripteurs sont présentées dans la figure 3 où l'on peut voir que leurs valeurs sont similaires dans des régions de même type. Cela montre que notre descripteur distingue avec succès des régions différemment texturées. La figure 4 montre l'effet de l'échelle r sur les descripteurs. De petites échelles (b) préservent les contours mais créent des descripteurs très bruités. A l'opposé, de grandes échelles décrivent bien des régions homogènes texturellement mais ne préservent pas les transitions nettes entre textures qui peuvent apparaître dans les images. Ceci est souligné dans l'image (c) où la transition nette entre les arbres et le ciel est lissée quand les descripteurs sont calculés sur un gros voisinage. Ce phénomène est parfaitement normal puisque pour ces pixels de transition, des caractéristiques du ciel et des arbres sont mélangées lors du calcul du descripteur, ce qui tend à représenter ce mélange comme une troisième texture. Cependant, ceci est problématique pour un transfert de couleur car ces descripteurs inexacts ont tendance à produire des halos de couleur autour des contours. Notez qu'il n'est pas possible d'intégrer simplement la luminance dans la fonction de poids w_r (comme dans le filtre bilatéral par exemple). En effet, cela empêcherait les textures très contrastée d'être décrites correctement car elles seraient fragmentées en plusieurs morceaux de luminance similaire. Pour notre application, nous devons satisfaire deux contraintes : des descripteurs homogènes à l'intérieur des régions texturées, et la préservation des contours nets de l'image.

4.2. Descente de gradient multi-échelle

Pour éviter que les transitions entre textures soient lissées, nous utilisons une descente de gradient multi-échelle. Intuitivement, cette descente de gradient propage les valeurs des descripteurs pertinents (calculés sur des régions homogènes) pour remplacer celles de descripteurs inexacts (impactés par une transition). Cette descente est guidée par la variance des descripteurs qui est faible lorsque les descripteurs sont calculés sur une région homogène et élevée autour des contours de l'image. Cette descente de gradient multi-échelle remplace les descripteurs ayant une forte variance par ceux calculés sur des régions uniformes.



(a) Descripteur $\mathcal{S}_{21}$ (b) Descente standard (c) Descente multi-échelle

Figure 5 : *Illustration de la descente de gradient.* *(a) Un zoom sur la transition ciel/arbres de l'image en figure 4. (b) Une descente de gradient guidée par la variance du descripteur calculé à grande échelle a tendance à affiner les contours (en haut), mais peut conduire à des descripteurs mal associés le long des transitions : le pixel rouge du ciel (en bas) est considéré comme appartenant aux arbres dans cet exemple. (c) Une descente de gradient effectuée progressivement à plusieurs échelles conserve mieux les transitions complexes entre textures. Le pixel rouge est maintenant bien associé au ciel.*

Concrètement, la variance d'un pixel $\mathbf{p}$ est calculée comme suit :

$$V_r(\mathbf{p}) = \| \frac{1}{W} \sum_{\mathbf{q} \in N_r^{\mathbf{p}}} (\mathcal{S}_r(\mathbf{q}) - \mathbf{v}_r)(\mathcal{S}_r(\mathbf{q}) - \mathbf{v}_r)^T w_r(\mathbf{p},\mathbf{q}) \|, \tag{2}$$

où $\mathcal{S}_r(\mathbf{p})$ est le descripteur du pixel $\mathbf{p}$ et $\mathbf{v}_r$ la moyenne pondérée des descripteurs appartenant au voisinage $N_r^{\mathbf{p}}$.

La descente de gradient déplace les descripteurs de chaque côté de la variance (c.a.d des contours de texture) et donc tend à affiner les contours des descripteurs. La figure 6 (en haut) montre le pseudo-code de la descente de gradient où la carte résultante contient les coordonnées où les descripteurs doivent être pris pour chaque pixel. Le déplacement résultant est montré dans la ligne supérieure de la figure 5 où les descripteurs initiaux (a) sont déplacés vers les régions homogènes en suivant le gradient de la variance (b). Le résultat dépends bien sûr de l'échelle à laquelle les descripteurs sont calculés. A de larges échelles, les transitions entre textures sont lissées et en conséquence des descripteurs peuvent être incorrectement attribués à différentes régions autour de ces transitions. Ce phénomène est illustré dans la ligne inférieure de la figure 5 où le pixel rouge appartenant au ciel (a) est incorrectement associé à un descripteur d'arbres (b) après la descente de gradient. Notre solution pour préserver les changements de texture complexes avec des descripteurs à grande échelle est d'utiliser une descente de gradient multi-échelle où l'échelle du descripteur et de sa variance est augmentée progressivement pour guider le déplacement du descripteur initial (calculé à grande échelle).

La figure 6 (en bas) présente le pseudo-code de cette descente de gradient multi-échelle. L'idée est d'appliquer plusieurs descentes de gradient itérativement à des échelles augmentant progressivement (de la plus fine à la plus large), afin de déplacer les pixels dans des régions homogènes tout en préservant les transitions complexes entre textures. A de fines échelles, la descente préserve bien les contours, mais arrive rapidement à des minima locaux. Augmenter progressivement l'échelle permet de sélectionner petit à petit des pixels de plus en plus loin des transitions complexes, assurant que les descripteurs restent cohérents. En pratique, le nombre d'itérations utilisé pour une échelle donnée est fixé à la taille du voisinage (de petites et grandes échelles permettent respectivement de petits et grands déplacements). Il est utile de préciser que même si des descripteurs à petite échelle sont utilisés pour calculer la variance, les coordonnées résultantes après descente sont uniquement utilisées pour modifier des descripteurs calculés à grande échelle. Le déplacement final est illustré dans la figure 5 (c). Le descripteur finalement obtenu (en haut) préserve plus précisément les transitions complexes entre textures. Le pixel rouge (en bas) se voit maintenant attribuer les valeurs d'un descripteur calculé dans une région homogène du ciel.

4.3. Filtre bilatéral non-normalisé

La descente de gradient assure une capture fine des propriétés texturelles autour de chaque pixel, même proche d'une transition entre textures. Cependant ces descripteurs peuvent toujours contenir des variations n'apparaissant pas dans l'image originale. Ces variations apparaissent typiquement autour de transitions en forme de U (comme dans la partie gauche de la figure 5 (c)) ou quand une région ne peut pas être correctement décrite par son contenu texturel (comme des lignes fines sur un fond uniforme). Cela doit être évité car toute variation dans les descripteurs peut créer des changements de couleur durant le transfert de couleur ou la colorisation. A cet effet, nous filtrons les descripteurs dans une dernière étape à l'aide d'un filtre préservant les contours afin de coller parfaitement à la structure de l'image. Pour cela nous adaptons le filtre bilatéral non-normalisé [APH*14] de manière à ce qu'il lisse itérativement les descripteurs en fonction des variations de luminance. Ce filtre est simple, efficace et introduit très peu de halos autour des contours. Cependant n'importe quel autre filtre préservant les

Descente de gradient

Entrée : carte de coordonnées M, carte de variance V, nombre d'itérations n
for all pixels **p do**
  for $i = 1$ to n **do**
    $M(\mathbf{p}) \leftarrow M(\mathbf{p}) + \nabla V(M(\mathbf{p}))$
  end for
end for
Return M

Descente de gradient multi-échelle

Initialiser M avec les coordonnées des pixels
Calculer $\boldsymbol{\mathcal{S}}_{r_{max}}$ avec l'équation. 1
for $r = 1$ to r_{max} **do**
  Calculer V_r avec l'équation. 2
  $M \leftarrow$ Descente de gradient(M, V_r, r)
end for
for all pixels **p do**
  $\boldsymbol{\mathcal{S}}_{r_{max}}(\mathbf{p}) \leftarrow \boldsymbol{\mathcal{S}}_{r_{max}}(M(\mathbf{p}))$
end for

Figure 6 : *Algorithme de descente de gradient multi-échelle.*

contours pourrait être utilisé [TM98, HST13, PM90]. Concrètement, nous utilisons le filtre bilatéral non-normalisé comme suit :

$$\boldsymbol{\mathcal{S}}^{ubf}(\mathbf{p}) = \boldsymbol{\mathcal{S}}(\mathbf{p}) + \sum_{\mathbf{q} \in N\mathbf{p}} \frac{G_{\sigma_s}(\mathbf{q}-\mathbf{p}) G_{\sigma_l}(L(\mathbf{q})-L(\mathbf{p}))(\boldsymbol{\mathcal{S}}(\mathbf{q})-\boldsymbol{\mathcal{S}}(\mathbf{p}))}{\sqrt{2\pi\sigma_s^2}}, \tag{3}$$

où $G_\sigma(\mathbf{x}) = exp(\frac{-\|\mathbf{x}\|^2}{2\sigma^2})$ est un noyau Gaussien standard. σ_s et σ_l contrôlent respectivement l'influence de la distance spatiale et de la variation de luminance. En pratique, l'équation 3 est appliquée itérativement avec de faibles valeurs de σ_s et σ_l (généralement 2 et 0.05) afin de diffuser les descripteurs correctement sur de larges voisinages. La figure 7 montre l'effet de ce filtrage sur une région problématique où les descripteurs ne suivent pas bien la frontière de la palme (a). Le filtre bilatéral non-normalisé recrée précisément les bords de la palme, comme montré en (b). La dernière image (c) montre le résultat du filtrage appliqué directement aux descripteurs initiaux (sans descente de gradient). Dans ce cas, les halos sont propagés dans les régions et impactent les descripteurs.

5. Edition locale de couleurs

Une fois des descripteurs fiables obtenus, nous proposons de les utiliser pour manipuler les couleurs localement en définissant des fonctions de transfert basées sur les pixels ayant des descripteurs similaires entre l'image d'entrée et la référence.

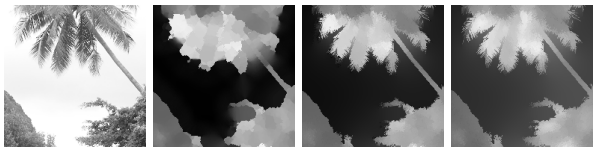


Figure 7 : ***Filtre bilatéral non-normalisé.*** *(a) Descripteur obtenu après descente de gradient. (b) Le filtre bilatéral non-normalisé propage précisément les descripteurs en respectant les frontières de luminance. (c) Sans descente de gradient, les halos sont propagés à l'intérieur des régions et altèrent les descripteurs. Dans ces exemples, 2000 itérations sont utilisées avec* $\sigma_s = 2$ *et* $\sigma_l = 0.05$.

5.1. Similarité entre pixels

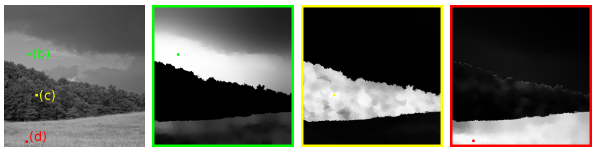
Notre mesure de similarité est basée sur la distance Euclidienne $L2$ entre deux descripteurs :

$$\mathcal{D}_{\sigma_d}(\mathbf{p}, \mathbf{q}) = \exp\left(\frac{-\|\boldsymbol{\mathcal{S}}(\mathbf{p}) - \boldsymbol{\mathcal{S}}(\mathbf{q})\|^2}{2\sigma_d^2}\right), \tag{4}$$

où $\boldsymbol{\mathcal{S}}(\mathbf{p})$ et $\boldsymbol{\mathcal{S}}(\mathbf{q})$ sont les descripteurs aux pixels $\mathbf{p}$ et $\mathbf{q}$ et σ_d est l'écart-type contrôlant comment la distance entre deux descripteurs contribue à la mesure de similarité. D'autres métriques auraient pu être utilisées comme détaillé dans [HCS*09, KEE13], mais nous n'avons pas remarqué de différences significatives entre ces métriques pour notre application. Des exemples de mesures de similarités sont présentées dans la figure 8, où les pixels (b), (c) et (d) sont comparés à tous les autres pixels de l'image d'entrée (a). On peut voir que les arbres, l'herbe et le ciel sont précisément séparés dans ces résultats.

5.2. Transfert de couleurs

Pour transférer des couleurs entre images, nous nous basons sur une modification d'histogramme locale entre les images d'entrée et de référence. La modification appliquée à chaque pixel est calculée à l'aide des pixels similaires à celui-ci, c'est-à-dire ayant une même texture sous-jacente. Le processus de transfert est basé sur une translation et un étirement de la distribution de couleurs dans un espace de couleur décorrélé, comme proposé initialement par Reinhard *et al.* [RAGS01]. Les images d'entrée et de référence sont donc transformées dans l'espace de couleur décorrélé



(a) (b) (c) (d)

Figure 8 : *Cartes de similarité.* *(a) Luminance de l'image d'entrée. Les pixels vert, jaune et rouge sont comparés à tous les autres à l'aide de l'équation 4 pour obtenir les cartes de similarités (b), (c) et (d). Cette mesure de similarité permet de discriminer avec précision les trois régions d'arbres, de ciel et d'herbe. Les similarités ont été calculées avec $\sigma_d = 1$ dans ces exemples.*

et perceptuellement uniforme CIE-Lab avant le transfert. La fonction de transfert suivante est ensuite appliqué à chaque canal $c \in \{L,a,b\}$ séparément :

$$\mathcal{T}_{\sigma_d}(\mathbf{p}) = \frac{std^{ref}(\mathbf{p})}{std^{in}(\mathbf{p})}\left(c^{in}(\mathbf{p}) - \mu^{in}(\mathbf{p})\right) + \mu^{ref}(\mathbf{p}), \quad (5)$$

où les indices "*in*", "*ref*" réfèrent aux images d'entrée et de référence respectivement. De leur coté, "μ", "*std*" sont les moyennes et écart-types pondérés, calculés comme suit en fonction des similarités avec le pixel $\mathbf{p}$ de l'image d'entrée :

$$\mu^{img}(\mathbf{p}) = \frac{1}{W}\sum_{\mathbf{q}} c^{img}(\mathbf{q})\mathcal{D}_{\sigma_d}(\mathbf{p}^{in},\mathbf{q}^{img})$$

$$std^{img}(\mathbf{p}) = \sqrt{\frac{1}{W}\sum_{\mathbf{q}} (c^{img}(\mathbf{q}) - \mu^{img}(\mathbf{p}))^2 \mathcal{D}_{\sigma_d}(\mathbf{p}^{in},\mathbf{q}^{img})},$$

où $img \in \{in, ref\}$ et W est le facteur de normalisation : $W = \sum_{\mathbf{q}} \mathcal{D}_{\sigma_d}(\mathbf{p}^{in},\mathbf{q}^{img})$.

Un exemple de transfert est montré en figure 9 (en haut) où l'on peut observer l'effet du paramètre σ_d. Quand σ_d est faible, les couleurs sont transférées uniquement entre régions très similaires, comme entre les mers ou les cieux des images d'entrée et de référence. Quand σ_d augmente, les régions considérées sont de plus en plus larges, créant un résultat de plus en plus proche d'un transfert global comme [RAGS01].

5.3. Colorisation

Les techniques de transfert d'histogramme ne peuvent pas être utilisées directement pour coloriser des images qui ne contiennent pas de canaux de chrominance. Dans ce cas, nous assignons simplement à chaque pixel de l'image d'entrée la chrominance moyenne de l'image de référence, pondérée par notre

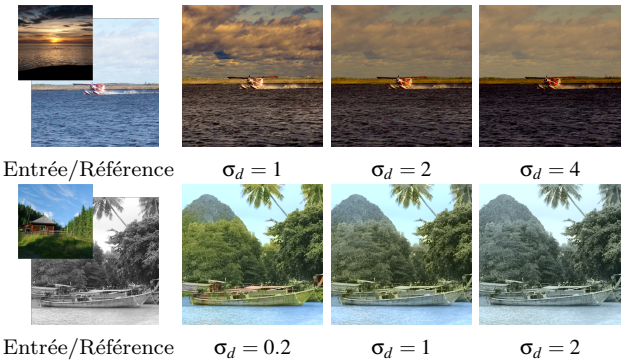


Figure 9 : *Impact de σ_d sur les fonctions de transfert.* *En haut : exemple de transfert de couleur. Quand σ_d augmente, de plus en plus de pixels sont considérés comme similaires, résultant en un transfert proche d'un transfert global d'histogramme. En bas : exemple de colorisation. Les couleurs étant obtenues à partir de la moyenne des pixels similaires de l'image de référence, augmenter σ_d a tendance à produire un résultat de plus en plus monochrome.*

mesure de similarité :

$$\mathcal{C}_{\sigma_d}(\mathbf{p}) = \frac{\sum_{\mathbf{q}} c^{ref}(\mathbf{q})\mathcal{D}_{\sigma_d}(\mathbf{p}^{in},\mathbf{q}^{ref})}{\sum_{\mathbf{q}} \mathcal{D}_{\sigma_d}(\mathbf{p}^{in},\mathbf{q}^{ref})}. \quad (6)$$

Notez que cette fonction de transfert est appliquée uniquement sur les canaux de chrominance (a et b), bien que la luminance pourrait aussi être modifiée en fonction de l'application souhaitée. Un exemple de colorisation est présenté en figure 9 (en bas). De grandes valeurs de σ_d ont tendance à moyenner les couleurs sur de grandes régions et ainsi créer des résultats pales et monochromes. Pour contrer cet effet, nous utilisons des valeurs faibles de σ_d pour la colorisation afin de ne moyenner que les régions de descripteurs très similaires.

5.4. Implémentation et performances

Nos fonctions de manipulation de couleurs sont intégralement implémentées sur GPU en Cuda. Tous les résultats présentés dans cet article ont été obtenus avec une carte NVIDIA Quadro 6000. En pratique, nous précalculons les descripteurs $\mathcal{S}$ pour les images d'entrée et de référence avant d'appliquer un transfert de couleur ou une colorisation. En fonction du nombre d'itérations choisi pour le filtre bilatéral non-normalisé, il faut entre 20 et 40 secondes pour obtenir les descripteurs des deux images. Néanmoins, les équations 5 et 6 nécessitent d'itérer sur tous les pixels de l'image d'entrée et de calculer pour chacun d'eux les

$\tau = 0$, 20min $\tau = 0.001$, 40s $\tau = 0.01$, 10s $\tau = 0.1$, <1s

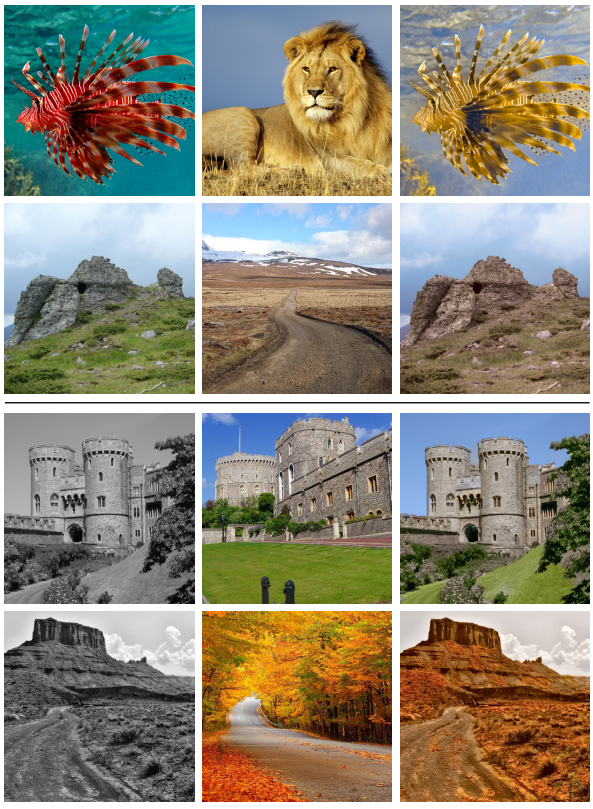
Figure 10 : *Effet de l'optimisation.* *Résultats du transfert de couleur pour des valeurs de τ croissantes, pour des images de taille 512×512. Plus τ est faible, plus l'algorithme est rapide et le risque d'obtenir des artefacts dus à la quantification augmente. Dans cet exemple, $\tau = 0.1$ permet un transfert en temps réel pouvant être utilisé pour explorer efficacement l'espace des résultats possibles malgré les artefacts visibles.*

similarités avec toute l'image de référence afin d'obtenir les moyennes et écart-types pondérés. Le nombre de calculs est donc conséquent et le transfert est lent, environ 20 minutes pour des images 512×512.

Pour obtenir des temps de calcul raisonnables, nous proposons de quantifier les similarités à l'aide d'une distance τ définie par l'utilisateur, contrôlant en dessous de quelle distance deux descripteurs sont considérés comme égaux. Si l'on considère un pixel d'entrée $\mathbf{p}$, tous les autres pixels $\mathbf{p}_i$ tels que $\mathcal{D}_{\sigma_d}(\mathbf{p}, \mathbf{p}_i) < \tau$ sont manipulés à l'aide de la même fonction de transfert. De cette façon, augmenter τ réduit le nombre d'itérations nécessaires pour obtenir le résultat. L'effet de cette optimisation est visible dans la figure 10, où l'on arrive à de gros gains de temps sans impact visuel sur le résultat. De grandes valeurs de τ ont tendance à produire des artefacts dus à la quantification, mais peuvent être utilisées pour explorer interactivement l'espace des résultats possibles.

En résumé, l'utilisateur peut jouer sur les paramètres suivants pour obtenir les résultats qu'il souhaite :

— r_{max} contrôle la taille de la fenêtre sur laquelle les descripteurs sont calculés et définit donc l'échelle à laquelle les textures sont analysées. En général, nous avons remarqué que $r_{max} = 21$ marche bien pour des images naturelles de taille 512×512.
— σ_s et σ_l contrôlent respectivement l'influence des distances spatiales et des variations de luminance lors du filtrage des descripteur avec le filtre bilatéral non-normalisé. Tous les résultats de cet article ont été calculés avec $\sigma_s = 2$ et $\sigma_l = 0.05$. Le nombre d'itérations utilisé pour ce filtrage dépend de la complexité des transitions entre textures. Nous utilisons typiquement 500 itérations pour nos résultats.
— σ_d contrôle comment le poids entre deux pixels est influencé par leur distance dans l'espace des descripteurs. En pratique, nous utilisons $\sigma_d = 1$ et $\sigma_d = 0.2$ pour la plupart des résultats de transfert de couleurs et de colorisation respectivement.
— τ contrôle le seuil de quantification. Généralement, une valeur de 0.01 permet un transfert assez rapide, sans artefacts visibles.



Entrée Référence Notre Résultat

Figure 12 : *Résultats de transfert et colorisation.* *Différentes couleurs sont associées à différentes régions en fonction de leur contenu texturel.*

6. Résultats

Les résultats et les comparaisons présentées dans cet article ainsi que les matériaux supplémentaires ont tous été calculés avec les paramètres par défaut donnés dans la section précédente.

6.1. Résultats du transfert de couleurs

La figure 11 (en haut) présente les résultats de notre transfert de couleurs comparé avec d'autres méthodes de l'état de l'art. Les résultats de [RAGS01, XM06]

Figure 11 : ***Comparaison avec les méthodes précédentes.*** *Le haut et le bas comparent nos résultats de transfert de couleur et de colorisation respectivement avec des méthodes précédentes de l'état de l'art. Se référer au texte pour plus de détails.*

ont été calculés avec notre implémentation de leur méthode. Les résultats de [PKD07,PR11] ont été produits avec le code disponible sur la page web des auteurs, nous avons utilisé 100% de ressemblance pour [PR11]. Les résultats de [HLMCB15] ont été pris sur la page des auteurs et ont dirigé notre choix d'images.

Ces résultats montrent en premier lieu que les approches globales [RAGS01, PKD07, XM06] ont tendance à produire des couleurs saturées dues à l'étirement de l'histogramme de couleur d'entrée. De plus, ces transferts globaux ont tendance à mettre en correspondance les régions de couleurs et luminance similaires, c'est pourquoi ils ne peuvent transférer entre régions de même contenu texturel si elles ont des grosses différences de luminance ou de couleurs. Ceci est illustré dans le dernier exemple où la couleur orangée des bâtiments est transférée dans le ciel de l'image d'entrée.

Les approches locales basées sur des informations de couleur [PR11, HLMCB15] créent de meilleurs résultats, mais n'arrivent pas non plus à mettre en correspondance les régions de même contenu texturel car elles définissent les régions en fonction de leur distribution de couleurs. Notre approche mets ces régions en correspondance avec succès, comme montré dans le troisième exemple où le jaune du champ de fleurs de la référence est bien transféré dans l'herbe de l'entrée ; ou dans le quatrième exemple où les bâtiments sont bien mis en correspondance dans les deux images, transférant le orange de la référence dans ceux d'entrée. La figure 12 montre deux exemples supplémentaires où la mise en correspondance entre régions est effectuée avec succès grâce à nos descripteurs.

6.2. Résultats de colorisation

La figure 11 (en bas) compare les résultats de notre colorisation à d'autres méthodes de l'état de l'art. Les résultats de [WAM02, CHS08, GCR*12] ont été pris dans l'article [GCR*12]. Les résultats de [BT12] ont été calculés à l'aide du code fourni par les auteurs, avec les paramètres par défaut suggérés dans leur code.

Ces résultats montrent que la méthode de [WAM02] basé sur la luminance ne marche pas quand les images d'entrée sont trop complexes : plusieurs régions avec la même luminance se voient attribuer la même couleur, comme le bâtiment et le ciel du premier exemple. La méthode de [CHS08] utilise des descripteurs SURF et des filtres de Gabor qui sont très discriminants, permettant une bonne colorisation quand les images d'entrée et de référence ont un contenu identique ou très similaire. Cependant, ils doivent couper les bords de l'image et ont tendance à créer des taches de couleurs dans leurs résultats. La méthode de [GCR*12] produit de meilleurs résultats à l'aide de descripteurs plus robustes, toutefois ils ont du mal à distinguer entre des régions intriquées comme le ciel et les nuages des premier et quatrième exemples, ou la rivière et la terre dans le quatrième exemple. Enfin, la méthode de [BT12] a tendance à créer des halos à cause de la fenêtre utilisée dans le calcul des descripteurs.

Comme on peut le voir dans la dernière colonne, notre approche colorise avec succès les régions de textures similaires : les couleurs du ciel, des nuages, de la végétation, de la montagne ou des bâtiments des images de référence sont transférées avec succès dans les images d'entrée. La figure 12 présente deux résultats supplémentaires avec une séparation claire entre les régions de l'image d'entrée et des associations de couleurs pertinentes avec l'image de référence.

6.3. Combiner colorisation et transfert

Etant donné que notre pipeline est le même pour la colorisation et le transfert de couleur, nous pouvons facilement combiner les deux avec une image en niveaux de gris en entrée en y ajoutant de la chrominance via colorisation, tout en modifiant sa luminance avec un transfert de luminance uniquement. Les résultats de cette approche sont présentés dans la figure 13 : cette combinaison peut produire un résultat plus proche du style de la référence qu'une colorisation classique avec une image en niveaux de gris en entrée. Comparé au résultat d'un transfert de couleur (transférant aussi la luminance), on peut voir que le transfert de couleurs reste plus riche en couleurs car la chrominance de l'image d'entrée est aussi utilisée durant le transfert. Cependant ce transfert est plus restrictif car il nécessite une image en couleurs en entrée.

7. Discussion et travaux futurs

Dans cet article, nous avons présenté un pipeline générique pour effectuer du transfert de couleurs ou de la colorisation. Nos descripteurs préservant les contours caractérisent précisément les régions de même contenu texturel entre les images, tout en étant robuste aux transitions entre textures. Il permet d'appliquer transfert de couleurs et colorisation localement entre régions similaires texturellement entre les images d'entrée et de référence. Notre méthode souffre de deux limitations principales décrites ci-dessous.

(1) Du coté de la colorisation, les images d'entrée et de référence doivent être assez similaires pour produire un résultat cohérent. Si une région de l'image d'entrée n'a pas de correspondance dans l'image de référence, la fonction de similarité (basée sur une distance Gaussienne) à tendance à attribuer le même poids à tous les pixels, produisant un résultat monochrome.

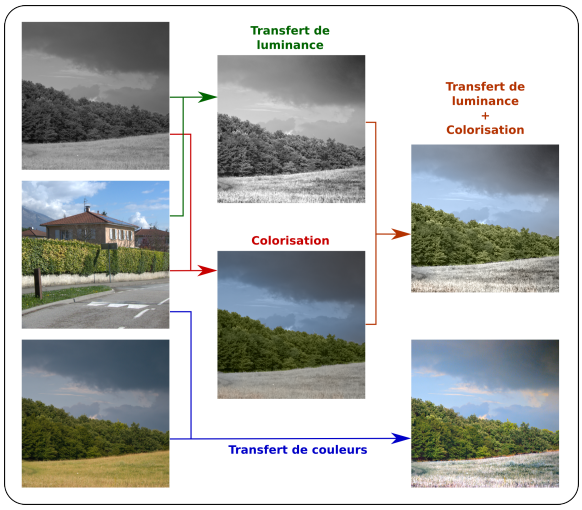


Figure 13 : *Combinaison de la colorisation et d'un transfert de luminance.* *Notre algorithme permet d'appliquer facilement une combinaison de colorisation et transfert de luminance. Cette combinaison permet d'effectuer un transfert de style entre l'entrée et la référence plus précis qu'une simple colorisation. Bien qu'il soit moins coloré qu'un transfert de couleur, ce résultat ne nécessite qu'une image en niveaux de gris en entrée. Dans ces résultats,* $\sigma_d = 0.5$.

Ceci est équivalent à augmenter σ_d pour cette région, comme illustré dans la figure 9 (en bas à droite). Ce problème a aussi lieu lors du transfert de couleurs mais il est beaucoup moins visible car la moyenne et l'écart-type ne sont utilisés que pour modifier l'histogramme. Pour éviter cela, une possibilité pourrait être de détecter automatiquement les régions sans correspondances proches et demander à l'utilisateur d'aider le transfert en fournissant une image de référence plus spécifique.

(2) Les descripteurs proposés permettent de caractériser efficacement les régions texturées ainsi que leurs transitions, mais ne sont pas capables de détecter d'information sémantique de plus haut niveau comme des visages, des objets manufacturés ou un arrière-plan et un premier plan. Nos descripteurs peuvent être altérés par ces types d'objets, impactant la qualité des résultats de transfert et de colorisation. Une fois de plus, ceci est plus visible dans les résultats de colorisation comme présenté dans la figure 14. La couleur jaune obtenue dans la partie supérieure gauche de l'image est due aux câbles électriques qui sont associés au panneau de signalisation présent dans la référence. Les roues de la mobylette contiennent aussi de fines structures associées à la casquette de la fille de la référence, résultant en une couleur bleutée dans les roues alors

Entrée Référence Notre Résultat

Figure 14 : *Cas Problématique.* *L'information sémantique comme des objets manufacturés ou des visages peut altérer localement les descripteurs, entraînant une colorisation incohérente. Par exemple, les roues de la mobylette sont colorées en bleu alors que le fond est principalement rouge. On s'attendrait à ce que les deux régions reçoivent la même couleur.*

que le fond est rouge clair. Une solution pour diminuer ces problèmes pourrait être d'utiliser des descripteurs plus complexes, mais plus lents, contenant à la fois de l'information texturelle et sémantique.

Malgré ces limitations, nous pensons que nos descripteurs présentent une bonne base qui pourrait être utilisable dans d'autres applications comme le changement de dynamique, la décomposition d'image tout en préservant les structures ou la modification des couleurs de séquences vidéo.

Remerciements

L'image d'entrée utilisée dans les figures 2, 3, 4, 5, 8, 10, 13 et l'image de référence contenant la maison dans la figure 9 proviennent du site `freebigpictures.com`. Les comparaisons du transfert de couleurs ont été faites avec les images de [HLMCB15]. Les comparaisons de la colorisation ont été faites avec les images de [GCR*12].

Références

[AP10] An X., Pellacini F. : User-controllable color transfer. *Computer Graphics Forum* (2010). 3

[APH*14] Aubry M., Paris S., Hasinoff S. W., Kautz J., Durand F. : Fast local laplacian filters : Theory and applications. *ACM Trans. Graph.. Vol. 33*, Num. 5 (2014), 167 :1–167 :14. 3, 6

[Bal06] Balas B. J. : Texture synthesis and perception : using computational models to study texture representations in the human visual system. *Vision Research. Vol. 46*, Num. 3 (2006), 299–309. 2

[BSPP13] Bonneel N., Sunkavalli K., Paris S., Pfister H. : Example-based video color grading. *ACM Trans. Graph.. Vol. 32*, Num. 4 (2013), 39 :1–39 :12. 3

[BT12] Bugeau A., Ta V.-T. : Patch-based image colorization. In *Pattern Recognition (ICPR), 2012 21st*

International Conference on (2012), pp. 3058–3061. 3, 10, 11

[CHS08] Charpiat G., Hofmann M., Schölkopf B. : Automatic image colorization via multimodal predictions. In *Proc. ECCV* (2008), pp. 126–139. 3, 10, 11

[CLKL14] Cho H., Lee H., Kang H., Lee S. : Bilateral texture filtering. *ACM Trans. Graph.. Vol. 33*, Num. 4 (2014), 128 :1–128 :8. 3

[DX09] Dong Y., Xu D. : Interactive local color transfer based on coupled map lattices. In *Proc. Computer-Aided Design and Computer Graphics* (Aug 2009), pp. 146–149. 3

[GCR*12] Gupta R. K., Chia A. Y.-S., Rajan D., Ng E. S., Zhiyong H. : Image colorization using similar images. In *Proc. ACM Int. Conference on Multimedia* (2012), pp. 369–378. 3, 10, 11, 12

[HCS*09] Hong X., Chang H., Shan S., Chen X., Gao W. : Sigma set : A small second order statistical region descriptor. In *Proc. IEEE CVPR* (2009), pp. 1802–1809. 5, 7

[HLMCB15] Hristova H., Le Meur O., Cozot R., Bouatouch K. : Style-aware robust color transfer. In *Proc. Computational Aesthetics* (2015), pp. 67–77. 3, 10, 11, 12

[HST13] He K., Sun J., Tang X. : Guided image filtering. *IEEE Trans. Pattern Analysis and Machine Intelligence. Vol. 35*, Num. 6 (2013), 1397–1409. 3, 7

[ICOL05] Irony R., Cohen-Or D., Lischinski D. : Colorization by example. In *Proc. EGSR* (2005), pp. 201–210. 3

[JCT14] Jin S.-Y., Choi H.-J., Tai Y.-W. : A randomized algorithm for natural object colorization. *Computer Graphics Forum. Vol. 33*, Num. 2 (2014), 205–214. 3

[KCP15] Kuzovkin D., Chamaret C., Pouli T. : Descriptor-based image colorization and regularization. In *Proc. Computational Color Imaging Workshop* (2015), pp. 59–68. 3

[KEE13] Karacan L., Erdem E., Erdem A. : Structure-preserving image smoothing via region covariances. *ACM Trans. Graph.. Vol. 32*, Num. 6 (2013), 176 :1–176 :11. 3, 4, 5, 7

[LLW04] Levin A., Lischinski D., Weiss Y. : Colorization using optimization. *ACM Trans. Graph.. Vol. 23*, Num. 3 (2004), 689–694. 3

[LSZ12] Liu S., Sun H., Zhang X. : Selective color transferring via ellipsoid color mixture map. *Journal of Visual Communication and Image Representation. Vol. 23*, Num. 1 (2012), 173 – 181. 3

[PKD07] Pitié F., Kokaram A. C., Dahyot R. : Automated colour grading using colour distribution transfer. *Comput. Vis. Image Underst.. Vol. 107*, Num. 1-2 (2007), 123–137. 2, 10, 11

[PM90] Perona P., Malik J. : Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Analysis and Machine Intelligence. Vol. 12*, Num. 7 (1990), 629–639. 3, 7

[PR11] Pouli T., Reinhard E. : Progressive color transfer for images of arbitrary dynamic range. *Computers & Graphics. Vol. 35*, Num. 1 (2011), 67–80. 3, 10, 11

[RAGS01] Reinhard E., Ashikhmin M., Gooch B., Shirley P. : Color transfer between images. *IEEE Comput. Graph. Appl.. Vol. 21*, Num. 5 (2001), 34–41. 2, 7, 8, 9, 10, 11

[TJT05] Tai Y.-W., Jia J., Tang C.-K. : Local color transfer via probabilistic segmentation by expectation-maximization. In *Proc. IEEE CVPR* (2005), vol. 1, pp. 747 – 754 vol. 1. 3

[TM98] Tomasi C., Manduchi R. : Bilateral filtering for gray and color images. In *Proc. ICCV* (1998), pp. 839–. 3, 7

[TPM06] Tuzel O., Porikli F., Meer P. : Region covariance : A fast descriptor for detection and classification. In *Proc. ECCV* (2006), pp. 589–600. 4

[WAM02] Welsh T., Ashikhmin M., Mueller K. : Transferring color to greyscale images. *ACM Trans. Graph.. Vol. 21*, Num. 3 (2002), 277–280. 3, 10, 11

[XDG10] Xia G.-S., Delon J., Gousseau Y. : Shape-based invariant texture indexing. *International Journal of Computer Vision. Vol. 88*, Num. 3 (2010), 382–403. 3

[XHJF12] Xu Y., Huang S., Ji H., Fermüller C. : Scale-space texture description on sift-like textons. *Computer Vision and Image Understanding. Vol. 116*, Num. 9 (2012), 999 – 1013. 3

[XM06] Xiao X., Ma L. : Color transfer in correlated color space. In *Proc. ACM Int. Conf. on Virtual Reality Continuum and its Applications (VRCIA)* (2006), pp. 305–309. 2, 9, 10, 11

[XYXJ12] Xu L., Yan Q., Xia Y., Jia J. : Structure extraction from texture via relative total variation. *ACM Trans. Graph.. Vol. 31*, Num. 6 (2012), 139 :1–139 :10. 3

[YJ*93] Yellott Jr J. I., et al. : Implications of triple correlation uniqueness for texture statistics and the julesz conjecture. *JOSA A. Vol. 10*, Num. 5 (1993), 777–793. 2