

Université de Montréal

Correction interactive de couleur par association

par

Christiane Landry

Département d'informatique et de recherche opérationnelle

Faculté des arts et des sciences

Mémoire présenté à la faculté des études supérieures

en vue de l'obtention du grade de

Maître ès sciences (M.Sc.)

en informatique

Février 1999

© Christiane Landry, 1999

Université de Montréal
Faculté des études supérieures

Ce mémoire de maîtrise intitulé

Correction interactive de couleur par association

présenté par
Christiane Landry

a été évalué par un jury composé des personnes suivantes :

Président : Jean Meunier

Directeur de recherche : Pierre Poulin

Membre : Pierre McKenzie

Sommaire

Les images font partie de presque tous les champs de l'activité humaine. Il est cependant parfois nécessaire de modifier une ou plusieurs images pour obtenir l'image désirée. Les changements fréquemment utilisés relèvent du domaine des couleurs. Plusieurs éditeurs permettant de modifier les couleurs d'une image appliquent plusieurs transformations de base les unes à la suite des autres. Cette série de transformations peut être difficile à trouver et le procédé n'est pas nécessairement très intuitif.

Nous avons développé une nouvelle approche interactive qui procure, dans certains cas, une alternative à l'approche traditionnelle. Elle se base sur le fait que l'utilisateur a souvent une idée assez précise des couleurs qu'il veut remplacer par d'autres. Nous lui demandons donc d'associer les couleurs à modifier avec les couleurs voulues dans le résultat final. Celles-ci peuvent être choisies à l'aide d'une palette de couleurs ou d'une ou de plusieurs images. Ces associations servent de contraintes pour calculer une transformation permettant la modification des couleurs vers celles désirées. Pour trouver cette transformation, nous utilisons la méthode des moindres carrés. Nous avons exploré cinq types de transformations : linéaire en RGB, quadratique en RGB, linéaire en (Luv) , linéaire en (L) et (uv) , et linéaire en (L) et cubique en (uv) .

Dans certains cas, la sélection de plusieurs couleurs peut être fastidieuse. Nous avons alors développé des méthodes d'association automatique de régions. Au lieu d'associer des couleurs une à une, l'utilisateur met en correspondance deux régions, indiquant ainsi qu'il veut que les couleurs de la première région se transforment en celles de la seconde. Ces méthodes choisissent les associations de couleurs parmi les deux régions. Les méthodes d'association de régions développées sont la méthode statistique et la méthode vectorielle. Ensuite, nous exposons une façon de combiner plusieurs associations de régions.

Nous présentons également un prétraitement qui améliore la transformation dans les zones où les couleurs sont saturées. Ce prétraitement extrapole les couleurs saturées selon le modèle de réflexion dichromatique afin d'obtenir une approximation plus satis-

faisante de ces couleurs pour les transformations.

Mots-clés :

Infographie 2D, éditeur de couleurs, transformation de couleurs, association de couleurs, association de régions, moindres carrés, extrapolation de couleurs saturées, modèle de réflexion dichromatique, RGB, CIE LUV

Table des matières

Remerciements	xv
Remarque préliminaire	xvi
1 Introduction	1
1.1 Applications	1
1.2 Problématique	3
1.3 Contenu	5
2 Les couleurs	7
2.1 Espaces de couleurs	8
2.2 Interfaces pour changer les couleurs	13
2.3 Transformation de gamut	16
3 Méthode d'évaluation des résultats	24
3.1 RGB	24
3.2 LUV	25
3.3 L'importance de la variance	26
3.4 Calcul de l'erreur sur une partie de l'image	28
4 Association des couleurs par l'utilisateur	30
4.1 Méthode d'association	31
4.2 Transformations dans l'espace RGB	32
4.2.1 Linéaire	34
4.2.2 Quadratique	39
4.3 Transformations dans l'espace LUV	46
4.3.1 Linéaire en (Luv)	46

4.3.2	Linéaire en (L) et (uv)	49
4.3.3	Linéaire en (L) et cubique en (uv)	52
4.4	Résultats comparatifs des transformations précédentes	55
5	Association automatique des couleurs	70
5.1	Méthodes d'association de régions	71
5.1.1	Méthode statistique	72
5.1.2	Méthode vectorielle	76
5.2	Transformation de plusieurs associations de régions	82
5.2.1	Application locale	83
5.2.2	Application globale	86
6	Correction de couleurs saturées	90
6.1	Modèle de réflexion dichromatique	91
6.2	Extrapolation des couleurs saturées	95
6.3	Exemples	97
7	Conclusion	105
7.1	Améliorations et travaux futurs	107
	Bibliographie	110

Liste des tableaux

4.1	Résultats de la transformation linéaire pour une rotation.	38
4.2	Résultats de la transformation linéaire pour un cisaillement avec changement d'échelle.	40
4.3	L'influence de la sélection des points pour les fig. 4.2 et 4.3 avec une transformation linéaire.	40
4.4	Résultats de la transformation quadratique pour une rotation.	42
4.5	Résultats de la transformation quadratique pour un cisaillement avec changement d'échelle.	43
4.6	Résultats de la transformation quadratique pour un gamma.	45
4.7	L'influence de la sélection des points pour les fig. 4.2, 4.3 et 4.6 avec une transformation quadratique.	45
4.8	L'influence de la sélection des points pour les fig. 4.2 et 4.7 avec une transformation linéaire en (Luv)	49
4.9	Approximation des angles maximaux de rotation.	52
4.10	L'influence de la sélection des points pour les fig. 4.2 et 4.7 avec une transformation linéaire en (L) et (uv)	52
4.11	Résultats des différentes transformations pour l' <i>inconstance de la caméra</i>	54
4.12	L'influence de la sélection des points pour les fig. 4.7 avec une transformation linéaire en (L) et cubique en (uv)	54
4.13	L'influence de la sélection des points pour la fig. 4.2 : <i>rotation</i> pour toutes les transformations.	56
4.14	L'influence de la sélection des points pour la fig. 4.3 : <i>cisaillement</i> pour toutes les transformations.	58
4.15	L'influence de la sélection des points pour la fig. 4.13 en haut : <i>compression</i> pour toutes les transformations.	59

4.16	L'influence de la sélection des points pour la fig. 4.6 : <i>gamma</i> pour toutes les transformations.	60
4.17	L'influence de la sélection des points pour la fig. 4.7 : <i>inconstance de la caméra</i> pour toutes les transformations avec 50 pts.	61
4.18	L'influence de la sélection des points pour la fig. 4.7 : <i>inconstance de la caméra</i> pour toutes les transformations avec 20 pts.	62
4.19	L'influence de la sélection des points pour la fig. 4.13 en bas : <i>changement d'éclairage</i> pour toutes les transformations.	62
6.1	Résultats des transformations avec et sans extrapolation pour le <i>dégradé</i> . . .	98
6.2	Résultats des transformations avec et sans extrapolation pour l' <i>oeuf</i> . . .	100
6.3	Résultats des transformations avec et sans extrapolation pour la <i>sphère</i> . . .	100
6.4	Résultats des transformations avec et sans extrapolation pour la <i>loupe</i> . . .	101

Table des figures

1.1	Exemple de photos prises par une caméra inconstante.	2
1.2	Exemple de composition d'images sans traitement préalable.	3
2.1	Fonctions de correspondance de couleur $\bar{r}_\lambda, \bar{g}_\lambda$ et \bar{b}_λ	9
2.2	Fonctions de correspondance de couleur \bar{x}, \bar{y} et \bar{z}	9
2.3	Cône des couleurs visibles dans l'espace CIE XYZ.	11
2.4	Diagramme de chromaticités CIE.	11
2.5	L'espace CIE LUV.	12
2.6	Transformations à l'aide de barres de défilement.	14
2.7	Menu de l'éditeur de couleur de xv.	15
2.8	Exemple de gamuts de moniteur, de film et d'imprimante.	17
2.9	Fonctions de compression de Stone et Wallace.	20
3.1	Défaillance du classement par la distance en RGB.	27
3.2	Défaillance du classement par la distance en LUV.	27
4.1	Mise en correspondance de couleurs par l'utilisateur.	31
4.2	Résultats d'une transformation linéaire pour retrouver une rotation et une translation.	37
4.3	Résultats d'une transformation linéaire pour retrouver la combinaison d'un cisaillement et d'un changement d'échelle.	39
4.4	Résultats d'une transformation quadratique pour retrouver une rotation.	42
4.5	Résultats d'une transformation quadratique pour retrouver un cisaillement avec changement d'échelle.	43
4.6	Résultats d'une transformation quadratique pour retrouver une transformation gamma.	44

4.7	Résultats d'une transformation linéaire en (Luv) pour retrouver l'inconstance de la caméra.	47
4.8	Région (côté droit) où est calculée l'erreur pour la fig. 4.7.	48
4.9	Résultats d'une transformation linéaire en (Luv) pour retrouver la rotation.	48
4.10	Résultats d'une transformation linéaire en (L) et (uv) pour retrouver l'inconstance de la caméra.	50
4.11	Résultats d'une transformation linéaire en (L) et (uv) pour retrouver la rotation.	51
4.12	Résultat d'une transformation linéaire en (L) et cubique en (uv) pour retrouver l'inconstance de la caméra.	54
4.13	Deux nouveaux exemples.	55
4.14	Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la fig. 4.2.	57
4.15	Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la fig. 4.3.	58
4.16	Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la fig. 4.13 en haut.	59
4.17	Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la fig. 4.6.	60
4.18	Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la fig. 4.7.	61
4.19	Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la fig. 4.13 en bas.	63
4.20	Comparaison des résultats des transformations sur la rotation.	64
4.21	Comparaison des résultats des transformations sur le cisaillement.	65
4.22	Comparaison des résultats des transformations sur la compression.	66
4.23	Comparaison des résultats des transformations sur la transformation gamma.	67
4.24	Comparaison des résultats des transformations pour l'inconstance de la caméra.	68
4.25	Comparaison des résultats des transformations sur le changement d'éclairage.	69

5.1	Images où la mise en correspondance est difficile.	70
5.2	Mise en correspondance de régions par l'utilisateur.	71
5.3	Correspondance de régions avec la méthode statistique.	73
5.4	Correspondance de régions ayant des proportions différentes avec la méthode statistique.	74
5.5	Cube RGB montrant les associations lorsque les régions n'ont pas les mêmes proportions.	75
5.6	Correspondance de régions par la méthode statistique sur l'exemple de la <i>rotation</i>	76
5.7	Correspondance de régions par la méthode statistique pour un exemple où il y a peu de changements de couleurs.	77
5.8	Correspondance de régions avec la méthode vectorielle.	79
5.9	Correspondance de régions avec la méthode vectorielle avec des régions comprenant beaucoup de couleurs différentes.	80
5.10	Correspondance de régions ayant des proportions différentes avec la méthode vectorielle.	80
5.11	Correspondance de régions par la méthode vectorielle sur l'exemple de la <i>rotation</i>	81
5.12	Correspondance de régions par la méthode vectorielle sur l'exemple du <i>cisaillement</i>	81
5.13	Correspondance de régions par la méthode vectorielle pour un exemple où il y a peu de changements de couleurs.	82
5.14	Variation de la fonction $H(r, g, b)$ lorsque $n = 1$ et 5.	84
5.15	Correspondance de plusieurs régions disjointes avec l'application locale.	87
5.16	Correspondance de plusieurs régions non-disjointes avec l'application lo- cale.	87
5.17	Correspondance de plusieurs régions non-disjointes avec l'application glo- bale.	89
6.1	Réflexions de surface et interne.	92
6.2	Aspects géométriques de la scène.	93
6.3	Combinaison linéaire de vecteurs couleurs.	94
6.4	Groupements linéaires en forme de "T".	95

6.5	Extrapolation des couleurs saturées.	96
6.6	Correction des couleurs saturées sur le <i>dégradé</i>	99
6.7	Correction des couleurs saturées sur l' <i>oeuf</i>	102
6.8	Correction des couleurs saturées sur la <i>sphère</i>	103
6.9	Correction des couleurs saturées sur la <i>loupe</i>	104

À Nicole

Remerciements

Je remercie mes parents de leur support moral et pour m'avoir incitée à poursuivre mes études jusqu'à la maîtrise. Je remercie également ma mère de sa patience lors de la correction du français de ce mémoire.

Je remercie Bernard de son support technique et de ses encouragements dans les moments difficiles de la rédaction. Un gros merci, Bernard.

Je remercie tous les étudiants du laboratoire d'infographie de l'université de Montréal pour les bons moments passés ensemble.

Je remercie mon directeur Pierre Poulin pour sa disponibilité et ses bons conseils pour mes recherches. Enfin, j'ai bien aimé jaser avec lui de choses et d'autres.

Je remercie M. Debevec pour me permettre d'utiliser certaines de ses photos au chapitre 6.

J'ai pu faire ma maîtrise grâce à une bourse ÉS/A du CRSNG et du financement du FCAR à la fin de la bourse. Je suis très reconnaissante à ces deux organismes pour leur appui.

Plusieurs organismes ont contribué au financement des équipements disponibles au laboratoire d'infographie : le CRSNG, le FCAR, l'Université de Montréal et Taarna Studios, inc.

Remarque préliminaire

Dans ce mémoire, les résultats de nos travaux sont des images couleurs. Il est donc important que les couleurs des images ne soient pas altérées par l'impression. Puisqu'on ne peut pas garantir l'exactitude des couleurs imprimées, nous avons mis les résultats couleurs sur une page *Web*. L'adresse de la page est *[http ://www.iro.umontreal.ca/labs/infographie/theses/landry](http://www.iro.umontreal.ca/labs/infographie/theses/landry)* . Les quelques images couleurs apparaissant dans ce mémoire ne doivent pas être considérées comme exactes, elles ne donnent qu'une idée générale des résultats.

Chapitre 1

Introduction

Dans le monde de la communication, l'image est très importante. Que ce soit pour la télévision, la publicité ou dans le domaine des sciences, l'image fait partie de la communication moderne. Ne dit-on pas qu'une image vaut mille mots ? Pour créer une nouvelle image, on peut la fabriquer de toute pièce en prenant une photo ou en la synthétisant à l'aide d'un ordinateur. On peut aussi réutiliser des images déjà existantes en les modifiant ou en les combinant. Qu'elle soit faite pour cette occasion ou réutilisée, l'image peut ne pas correspondre exactement au résultat souhaité. Même les photos des plus beaux mannequins sont retouchées. Dans certains cas, il est difficile ou même impossible d'avoir l'image désirée sans apporter des modifications plus ou moins importantes. Il est souvent préférable et plus simple de modifier l'image ou une partie de l'image pour obtenir le résultat désiré.

Une manipulation importante d'une image concerne l'aspect des couleurs. Dans plusieurs situations, l'utilisateur veut seulement modifier les couleurs de l'image sans changer les autres aspects de la scène. La section suivante présente quelques exemples d'applications de modification de couleurs.

1.1 Applications

La modification des couleurs d'une image sans changer les autres aspects de la scène peut servir dans maintes applications. Juste à titre d'exemples, nous en nommons trois. Une première application est liée à la technologie des caméras digitales. Ces caméras ne sont pas encore parfaites. Certaines caméras sont inconstantes, c'est-à-dire que si vous

photographiez deux fois la même scène avec le même éclairage vous n'obtiendrez pas toujours le même résultat. La figure 1.1 montre deux photos prises avec la même caméra et la même source de lumière à une minute d'intervalle. Une des photos est beaucoup plus rosée¹ que l'autre. Pour corriger cette différence et calibrer les deux images, nous devons modifier les couleurs de la photo rosée pour la rendre comme l'autre photo. Une autre caractéristique de ces appareils est qu'ils effectuent automatiquement et sans contrôle de l'utilisateur des transformations sur les couleurs (balancement du blanc, choix de l'intervalle de luminance enregistrée, etc.). La calibration des couleurs de deux images peut aussi servir sur des vieux films ou photos décolorés et lors d'assemblage de séquences de films.

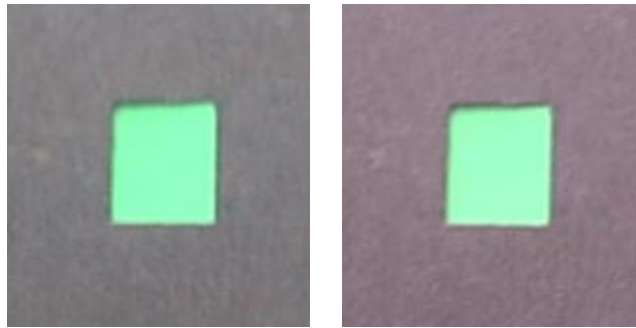


FIG. 1.1: Exemple de photos prises par une caméra inconstante avec un éclairage inchangé. Les images sont prises avec une caméra VPC G200 de Sanyo.

La modification des couleurs d'une image peut servir aussi dans la composition d'images [PD84] [SB96]. Lorsqu'on fait de la composition d'images, il faut surveiller attentivement l'éclairage dans les différentes images. Si l'objet à incorporer dans une image est illuminé par un éclairage trop différent de l'éclairage de l'image, cette différence sera nettement visible et l'objet n'aura pas l'air de faire partie de la scène. Dans l'image de la figure 1.2, le lion, le veau et l'agneau ne sont pas éclairés de la même façon. Avant l'incorporation de l'objet, il faut donc traiter ses couleurs pour que l'objet ait l'air de recevoir le même éclairage que le reste de la scène. Par exemple, si l'image d'une personne ensoleillée doit être incluse dans un paysage sous un ciel nuageux, le traitement préalable des couleurs de la personne est indispensable.

Dans une autre perspective, une image panoramique de 360° [Che95], peut être

¹ Comme indiqué dans la remarque préliminaire, les images apparaissent en couleur sur la page Web <http://www.iro.umontreal.ca/labs/infographie/theses/landry>.



FIG. 1.2: Exemple de composition d'images sans traitement préalable. Cette image provient de [MB95].

construite à partir d'une série de photos en superposant chaque photo. Les parties de la scène apparaissant dans plus d'une photo devraient avoir les mêmes couleurs. Dans la réalité, il y a des différences de couleurs. On doit uniformiser les couleurs d'une photo par rapport à l'autre de façon à ce qu'il n'y ait pas de discontinuité de couleur à la jonction des photos.

Ces applications ne sont pas les seules, mais elles ont un point en commun : on sait très précisément quelles couleurs sont désirées dans le résultat final. Ces couleurs pourraient être utilisées dans la transformation des couleurs de l'image à modifier.

1.2 Problématique

Il existe sur le marché plusieurs éditeurs de couleurs. Ces éditeurs permettent de modifier les couleurs de l'image ou d'une partie de l'image. Les modifications typiques se font par des transformations spécifiques comme une rotation dans l'espace couleur, une translation, une désaturation, un changement de contraste ou de brillance, etc. Elles permettent aussi à l'utilisateur de modifier la fonction de correspondance entre une couleur de l'image à modifier et son équivalent dans l'image résultante. Ces éditeurs offrent une série de transformations de base et qui, combinées ensemble, décuplent les possibilités de modifier les couleurs. Cependant, l'utilisateur peut éprouver des difficultés à déterminer la bonne combinaison de transformations de base pour obtenir le résultat désiré. Il faut choisir judicieusement les transformations à employer. Chaque transformation a un ou plusieurs paramètres. Ensuite, il faut utiliser ces transformations dans le bon ordre car certaines séries de transformations ne donnent pas le même résultat si elles sont agencées d'une autre façon. Lorsqu'on applique la dernière transformation choisie, il est assez

simple de comparer le résultat obtenu avec le résultat désiré. Mais à l'application d'une transformation intermédiaire, il est très difficile de déterminer si cette transformation est satisfaisante ou s'il faut modifier les paramètres. Si le résultat ne donne pas le résultat voulu, quelle étape est fautive ? Seule l'expérience peut guider l'utilisateur vers le résultat qu'il veut.

Une approche plus intuitive serait la bienvenue. Plusieurs transformations de base sont des transformations du premier degré dites “linéaires” comme la rotation, la translation, le changement d'échelle, le cisaillement, etc. Toutes les combinaisons de ces transformations sont des transformations linéaires. Au lieu de trouver tous les paramètres de chacune des transformations de base, il serait peut être plus simple de trouver les paramètres de la transformation linéaire qui englobe toutes les transformations de base. Cependant, ces paramètres sont encore moins intuitifs que ceux des transformations de base. Il nous faut plus d'informations sur le résultat désiré. Une façon intuitive d'effectuer ces transformations serait d'associer quelques couleurs de l'image à modifier aux couleurs qu'on voudrait qu'elles deviennent. Avec ces associations, il est très facile de trouver une transformation linéaire même complexe si elle existe.

Notre éditeur fonctionne selon ce principe. On utilise une seconde image ou une palette contenant un sous-ensemble des couleurs désirées dans le résultat final. Notre éditeur fait interactivement la correction des couleurs de l'image selon des associations de couleurs établies par l'utilisateur ou encore automatiquement. Les associations sont faites entre les couleurs de l'image de départ et les couleurs souhaitées dans le résultat final. Ces couleurs souhaitées peuvent provenir d'une ou de plusieurs images ou d'une palette de couleurs. Une fois les associations faites, l'éditeur calcule la “meilleure” transformation permettant de passer des couleurs à modifier aux couleurs souhaitées. Cette méthode est inverse aux méthodes utilisées par les éditeurs de couleurs traditionnels puisqu'au lieu de partir de l'image à modifier et d'appliquer des transformations jusqu'au moment où l'utilisateur est satisfait, on part d'un sous-ensemble des couleurs du résultat désiré et on calcule la transformation complexe permettant d'y parvenir. Le sous-ensemble des couleurs désirées peut provenir de une ou plusieurs images ou même d'une palette parce qu'on utilise seulement l'information sur la couleur. On ne tient pas compte de la position de ces couleurs dans les images ou la palette et encore moins des couleurs avoisinantes. Si on utilise une palette de couleurs, il faut que les couleurs

de la palette soient suffisamment variées pour ne pas transformer plusieurs couleurs de l'image de départ vers une seule couleur. L'utilisation d'une palette de couleurs appropriée permet de modifier les couleurs d'une image sans avoir besoin d'une seconde image. En associant bien les couleurs à modifier avec des couleurs de la palette, on obtient la même diversité de couleurs que dans l'image de départ mais avec les couleurs choisies dans la palette.

Notre éditeur est parfaitement adapté aux applications que nous avons décrites précédemment. Pour calibrer des images les unes par rapport aux autres, on choisit une des images comme image de référence. Le sous-ensemble des couleurs désirées est fourni par l'image de référence. Pour l'exemple d'une personne ensoleillée à inclure dans un paysage nuageux, il suffit de retrouver des couleurs apparaissant sur la personne telles qu'elles devraient être sous un ciel nuageux. Dans ce cas, il faut faire particulièrement attention aux ombres et aux "highlights" de la personne à inclure. Pour les images panoramiques, notre éditeur permet d'associer les couleurs qui devraient être identiques dans les régions superposées et ensuite d'ajuster les couleurs d'une photo par rapport à l'autre comme dans l'article de Pham et Pringle [PP95].

1.3 Contenu

Ce mémoire est divisé en sept chapitres. Le chapitre 2 présente les notions de base sur la couleur pour bien comprendre le reste du mémoire. Le chapitre 3 discute de notre façon d'évaluer les résultats d'un changement de couleur.

Le chapitre 4 aborde les associations de couleurs faites par l'utilisateur. L'utilisateur choisit un certain nombre de couleurs dans l'image de départ et les associe chacune avec une couleur de l'image objectif. Une fois ces associations faites, on peut transformer ces couleurs de plusieurs façons. Ce chapitre discute des diverses transformations qu'on a explorées dans le cadre de ce mémoire.

Le chapitre 5 présente les associations par régions. Au lieu d'associer des couleurs spécifiques entre elles, l'utilisateur choisit des régions. On compare deux méthodes pour sélectionner et apparier les couleurs de ces régions. Ce chapitre présente aussi l'influence de chaque région sur la transformation de l'image lorsqu'il y a plusieurs régions sélectionnées.

Le chapitre 6 discute d'une méthode pour extrapoler un gradient pour les couleurs

saturées d'une image, ce qui pourra être utile lors des transformations. Cette méthode permet de ramener la couleur saturée vers la couleur saturée qu'elle aurait si les couleurs extérieures étaient tronquées dans le sens du gradient au lieu d'utiliser les axes de l'espace couleur.

Finalement, nous concluons avec le dernier chapitre de ce mémoire.

Chapitre 2

Les couleurs

Les couleurs sont présentes partout où l'on regarde. Elles font partie intégrante de l'existence de la majorité d'entre nous. Mais toutes les personnes ne perçoivent pas les couleurs de la même façon. Si on demande à deux personnes “Quelle est cette couleur?”, l'une peut répondre “bleu ciel” alors que l'autre dirait plutôt “turquoise”. Il faut s'entendre sur une façon d'identifier les couleurs.

Pour cette raison, ce chapitre commence par aborder les espaces de couleurs permettant l'identification des couleurs visibles ou des couleurs affichables sur un moniteur. Ensuite, nous discuterons des possibilités qu'offrent les différentes interfaces pour modifier les couleurs d'une image. Les couleurs peuvent être affichées à l'aide de différents médiums tels un moniteur, une imprimante ou un film. Ces médiums ne peuvent pas afficher la même gamme de couleurs. Cette gamme de couleurs est appelée *gamut*. Puisque les couleurs d'une image affichées à l'écran ne sont pas toutes reproductibles par une imprimante, il faut transformer ces couleurs pour que l'image imprimée soit la plus près possible de l'image affichée. Ce procédé est nommé dans la littérature *la transformation de gamut*. Puisque la transformation de gamut est très proche des transformations de couleurs que nous faisons, la dernière section de ce chapitre lui sera consacrée.

Ce chapitre ne fait qu'un survol des connaissances sur les couleurs parce que le sujet est très vaste. Mais, il contient tout ce qui est nécessaire de savoir sur les couleurs pour bien comprendre les chapitres suivants.

2.1 Espaces de couleurs

Il y a plusieurs espaces de couleurs qui sont utilisés dans différents domaines. L'espace CIE XYZ sert en photométrie, CIE LUV en visualisation scientifique, CIE LAB dans les textiles, RGB pour les moniteurs, CMY pour l'impression, YIQ pour la télévision, HSV, HSI et HLS sont utilisés pour la sélection de couleurs, Munsell [Mun76] en psychologie et Ostwald [Ost31] en peinture. Il existe d'autres espaces encore, mais ceux-ci sont les principaux. Chaque domaine a inventé un espace facilitant ses applications. Les espaces les plus importants sont expliqués dans le livre de Foley et al. [FvDFH90]. Les espaces qui nous intéressent pour notre propos sont : CIE XYZ, RGB et CIE LUV¹.

CIE XYZ

L'œil humain perçoit les couleurs avec trois types de cônes différents. Les scientifiques ont associé chaque type de cônes à une longueur d'onde : courte à 438.1 nm, moyenne à 546.1 nm et longue à 700.0 nm. Ces longueurs d'onde correspondent respectivement à du bleu, du vert et du rouge. Ils ont déterminé empiriquement quelle quantité de chacune des trois couleurs il fallait additionner pour produire chaque longueur d'onde du spectre visible selon un observateur "moyen". La figure 2.1 indique ces quantités par les fonctions de correspondance de couleur \bar{r}_λ , \bar{g}_λ et \bar{b}_λ associées respectivement au rouge, au vert et au bleu.

Le processus pour obtenir ces fonctions est très simple. Ils projettent sur un écran la longueur d'onde qu'ils veulent obtenir. À côté, ils projettent dans la même région les trois couleurs associées aux trois longueurs d'onde de base. Ils permettent à l'utilisateur de modifier l'intensité des trois couleurs jusqu'à l'obtention d'une couleur équivalente selon l'utilisateur à la longueur d'onde voulue. Les valeurs négatives de la fonction du rouge signifient qu'il est impossible avec ces trois longueurs d'onde d'obtenir la longueur d'onde voulue. Pour pouvoir obtenir la couleur demandée, il a fallu ajouter à la couleur cible la quantité de rouge qui correspond à la partie négative de la courbe du rouge dans la figure.

En 1931, la *Commission Internationale de l'Éclairage* (CIE) a décidé d'utiliser trois

¹Nous présentons l'espace couleur CIE $L^*u^*v^*$. Pour alléger le texte, nous le nommerons LUV. Pour les mêmes raisons, nous appellerons les composantes L^* , u^* et v^* respectivement L , u et v .

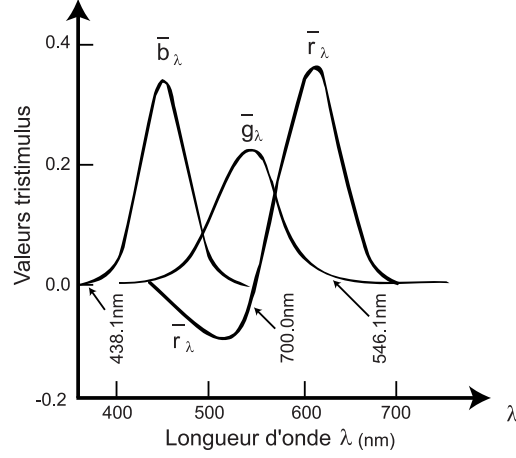


FIG. 2.1: Fonctions de correspondance de couleur \bar{r}_λ , \bar{g}_λ et \bar{b}_λ . Combinaison de trois longueurs d'onde pour obtenir toutes les longueurs d'onde du spectre visible. Ce graphique provient de [FvDFH90].

couleurs primaires pour remplacer le rouge, le vert et le bleu. Ces trois couleurs X , Y et Z ont été choisies parce que leurs fonctions sont positives pour toutes les longueurs d'onde du spectre visible. En faisant une combinaison linéaire avec ces trois couleurs, on peut obtenir une couleur équivalente à n'importe quelle longueur d'onde visible. Cependant, les couleurs X , Y et Z ne sont pas toutes visibles. La figure 2.2 montre les fonctions de correspondance de couleur \bar{x} , \bar{y} et \bar{z} associées aux couleurs X , Y et Z pour obtenir la longueur d'onde voulue. Pour avoir les valeurs exactes de ces fonctions, consultez le livre de Wyszecki et Stiles [WS82].

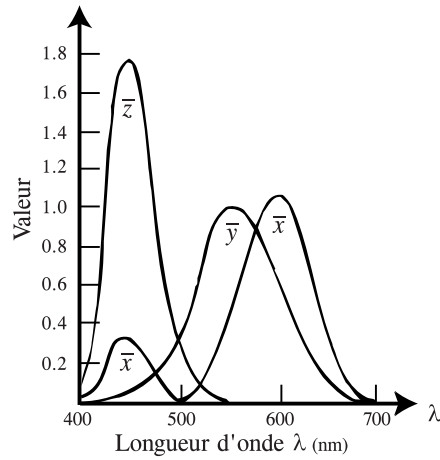


FIG. 2.2: Fonctions de correspondance de couleur \bar{x} , \bar{y} et \bar{z} . Combinaison des couleurs : X , Y et Z en fonction de la longueur d'onde. Ce graphique provient de [FvDFH90].

Dans l'espace XYZ, les couleurs visibles forment un cône qu'on peut voir à la figure 2.3. Cet espace est additif, c'est-à-dire que lorsqu'on superpose des couleurs on additionne les composantes de chaque couleur pour obtenir la composition de la couleur finale. Les valeurs de chromaticité CIE : x, y et z sont définies en normalisant X, Y et Z :

$$\begin{aligned} x &= \frac{X}{X + Y + Z} \\ y &= \frac{Y}{X + Y + Z} \\ z &= \frac{Z}{X + Y + Z}. \end{aligned} \tag{2.1}$$

Puisque $x + y + z = 1$, on n'indique que les valeurs de x et y . Les points (x, y, z) sont dans le plan $X + Y + Z = 1$ qu'on peut voir à la figure 2.3. Le diagramme de chromaticité CIE est la projection des valeurs x, y, z visibles sur le plan $Z = 0$. Le diagramme de chromaticité CIE est présenté à la figure 2.4. Toutes les couleurs ne se retrouvent pas dans ce diagramme puisque la dimension de la luminance n'est pas présente. Le diagramme de chromaticité possède quelques caractéristiques intéressantes. Toutes les couleurs pures (monochromatiques) du spectre du visible se retrouvent sur la courbe formant le fer à cheval dans le diagramme. Toutes les combinaisons linéaires de deux couleurs dans le diagramme se retrouvent sur la ligne reliant ces deux couleurs. Si l'on pondère positivement trois couleurs dans le diagramme, on peut produire toutes les couleurs comprises dans le triangle qui a pour sommets les trois couleurs de départ. Cette dernière caractéristique est très utile pour montrer la gamme de couleurs produites par un médium. Ce diagramme est très souvent utilisé pour comparer les différents gamuts des médiums.

L'espace CIE XYZ comprend toutes les couleurs visibles pour l'être humain. Cependant, pour s'adapter aux spécifications des moniteurs couleurs, des chercheurs ont inventé l'espace RGB.

RGB

Les moniteurs produisent les couleurs en excitant trois types de phosphores disposés de façon suffisamment rapprochée pour que l'oeil humain détecte seulement la combinaison des couleurs des trois phosphores. Pour choisir les couleurs des phosphores, deux critères ont été préconisés : la facilité d'obtenir la couleur avec du phosphore et que

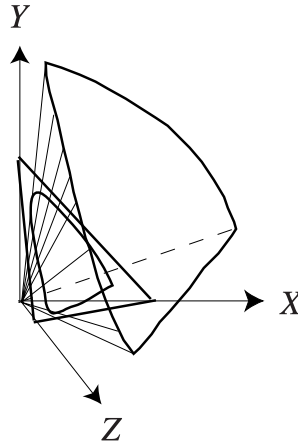


FIG. 2.3: Cône des couleurs visibles dans l'espace CIE XYZ. Ce graphique provient de [FvDFH90].

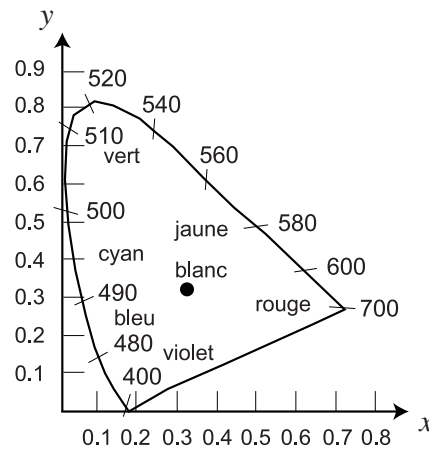


FIG. 2.4: Diagramme de chromaticités CIE. Le point représente l'emplacement du blanc dans le diagramme. Ce graphique provient de [FvDFH90].

l'ensemble des trois couleurs forme un espace de couleurs additif. Les couleurs choisies sont le rouge, le vert et le bleu. Puisqu'on peut régler l'intensité de chaque type de phosphores pour obtenir la couleur globale, il est plus simple d'utiliser un espace ayant ces trois couleurs comme axes. L'espace RGB forme un cube unitaire dont les axes orthogonaux sont R, G et B qui représentent respectivement le rouge, le vert et le bleu. Pour chaque axe, la valeur comprise entre 0 et 1 donne la proportion de l'intensité maximale de la composante. L'espace RGB est un espace additif.

Puisque l'encodage le plus courant est de diviser chaque composante en 256 niveaux d'intensités (un octet), l'intervalle de 0 à 255 en valeurs entières pour chaque axe est

aussi très utilisé. Cet espace est axé sur la couleur des phosphores des moniteurs. Comme la couleur des phosphores varie légèrement d'un moniteur à l'autre, les couleurs de cet espace dépendent du moniteur utilisé.

CIE LUV

Les deux espaces précédents ne sont pas perceptuellement uniforme, c'est-à-dire que si l'on prend deux paires de couleurs $(a, a + \delta)$ et $(b, b + \delta)$, les deux couleurs de chaque paire ne donneront pas nécessairement l'impression d'être séparées par la même distance. L'espace CIE LUV tente de remédier à cela. Quoique l'espace CIE LUV n'est pas exactement perceptuellement uniforme, c'est l'espace qui s'en approche le plus. L'espace LUV est représenté à la figure 2.5. Le L représente la luminance et u et v les composantes chromatiques.

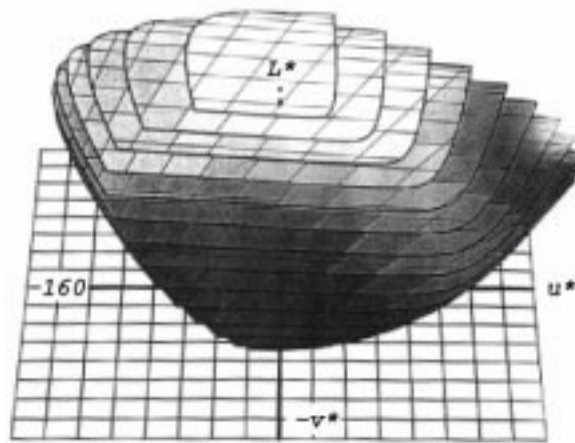


FIG. 2.5: L'espace CIE LUV. Ce graphique provient de [HRV97].

Nous utiliserons les espaces RGB et CIE LUV dans les chapitres suivants. L'espace RGB est utile puisqu'on travaille avec des moniteurs et l'espace CIE LUV est utilisé parce qu'il est presque perceptuellement uniforme. L'espace CIE LAB est également presque perceptuellement uniforme. Dans la littérature, les deux espaces sont aussi souvent utilisés l'un que l'autre. Nous avons privilégié l'espace CIE LUV parce qu'il a été conçu pour la visualisation scientifique contrairement à l'espace CIE LAB qui servait principalement dans le domaine des textiles. On peut passer de l'espace RGB à l'espace CIE LUV comme on le verra au chapitre 3. L'espace CIE XYZ sert d'espace intermédiaire entre ces deux espaces lors de la transformation. Avant de présenter ce

que nous avons fait en matière de transformation de couleurs, nous exposerons les possibilités des interfaces déjà existantes.

2.2 Interfaces pour changer les couleurs

Sur le marché, il existe plusieurs logiciels pour modifier des images. Ces logiciels permettent de changer la taille et l'orientation de l'image, de couper une partie de l'image, de filtrer l'image pour obtenir différents effets comme de rendre l'image plus floue ou de lui donner l'aspect d'une peinture. Certains peuvent faire de la composition d'images, c'est-à-dire prendre plusieurs portions d'images et les agencer pour former une nouvelle image. Ils permettent aussi de changer les couleurs d'une image. Cette section présente les changements de couleurs proposés par ces logiciels qui ne modifient pas la scène de l'image. Nous ne discuterons pas des filtres qui modifient à la fois la scène et les couleurs. Par exemple, un filtre qui donne l'aspect d'une peinture à une image modifie les couleurs, mais il ajoute aussi une texture de toile à l'image.

Certains logiciels sont axés sur l'édition des photos comme *PhotoDeluxe*. Une fois les photos numérisées, ils offrent toute une panoplie d'outils pour modifier l'aspect des photos. Pour les transformations de couleurs proprement dit, ils n'offrent pas un très vaste choix de transformations. Plusieurs transformations utilisent des barres de défilement ("slider") pour ajuster une composante d'un espace de couleurs. Par exemple, on peut ajuster la luminance, la teinte, la saturation ou le contraste d'une image. À la figure 2.6, ils proposent un balancement des couleurs de l'image en utilisant des barres de défilement pour augmenter/diminuer le rouge par rapport au cyan, le vert par rapport au magenta et le bleu par rapport au jaune. Ces transformations s'appliquent le plus souvent sur toute l'image et seulement un paramètre permet de varier la fonction dans chaque dimension. Ces transformations se font interactivement mais l'utilisateur doit essayer des combinaisons jusqu'à l'obtention d'un résultat satisfaisant.

Ensuite, il y a des transformations adaptées spécialement pour les photos. *PhotoDeluxe* offre une transformation pour enlever le problème des "yeux rouges" lorsque le flash de la caméra est mal orienté. Pour ce faire, l'utilisateur indique la région de la pupille et le logiciel diminue petit à petit la composante rouge de la région. La région doit être bien délimitée parce que la diminution du rouge où il ne faut pas rend les couleurs comme la peau plus verdâtres. *PhotoDeluxe* peut aussi transformer une image couleur

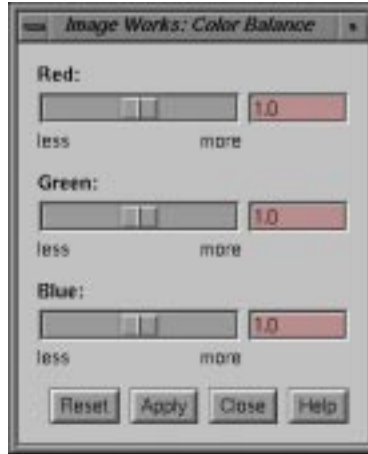


FIG. 2.6: Transformations à l'aide de barres de défilement. Ce menu provient de l'éditeur *Image Works*.

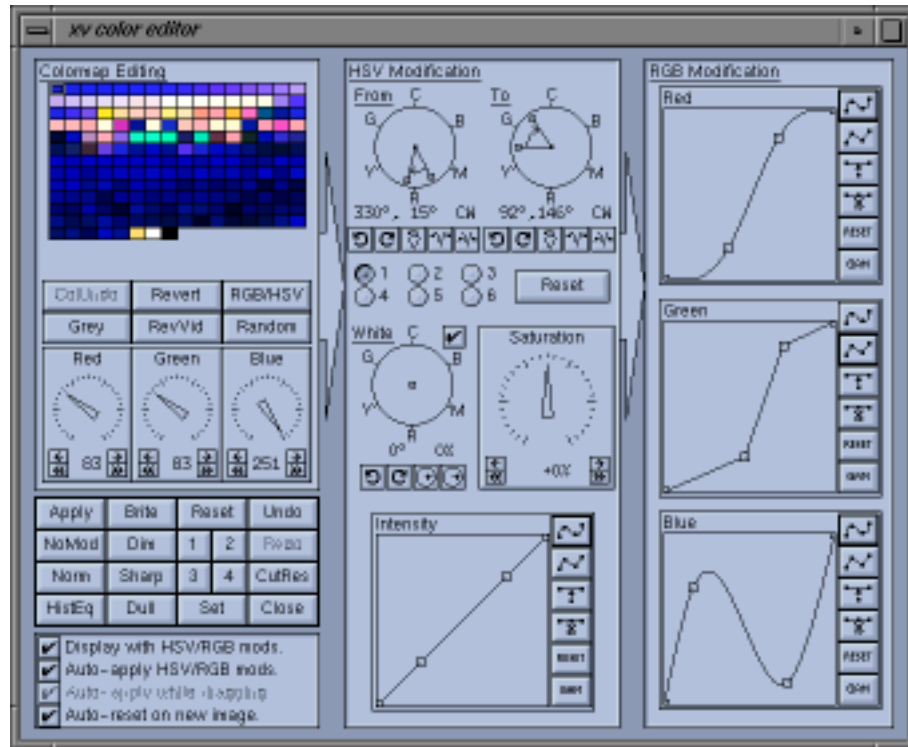
en gris en conservant seulement la composante de luminance L de la couleur.

PhotoDeluxe a aussi des filtres qui utilisent la version achromatique de l'image pour transformer la luminance vers une gamme de couleurs. Par exemple, un filtre permet de donner l'apparence d'un éclairage au néon après avoir transformé l'image en gris. Cette transformation, en plus de n'utiliser que la luminance, est très rigide puisqu'elle ne propose pas de paramètres pour varier la transformation à part la couleur du néon.

La plupart des logiciels permettent aussi à l'utilisateur de retoucher lui-même les couleurs à l'aide d'une palette de couleurs et de différentes brosses. Cette modification des couleurs n'est pas vraiment une transformation de couleurs puisqu'elle ne tient pas compte de la couleur originale du pixel modifié.

xv offre plus de souplesse et permet de modifier les fonctions elles-mêmes pour transformer les couleurs. La figure 2.7 montre le menu de l'éditeur de couleur. Dans la partie droite du menu, l'éditeur présente le graphique de la fonction de correspondance dans chaque dimension de l'espace utilisé. L'utilisateur peut modifier la fonction en déformant la courbe du graphique ou entrer un paramètre d'une fonction prédéfinie comme la fonction *gamma*. Encore une fois ce processus se fait interactivement et habituellement sur toute l'image. L'utilisateur modifie les fonctions tant qu'il n'a pas le résultat souhaité.

xv permet l'utilisation de carte de couleurs où toutes les couleurs d'une image sont remplacées par les couleurs de cette palette. La figure 2.7 donne un exemple de l'utilisation d'une carte de couleurs dans la partie gauche. L'utilisateur n'a qu'à changer une couleur pour une autre. Pour ce faire, il faut que l'image ait un nombre limité de

FIG. 2.7: Menu de l'éditeur de couleur de *xv*.

couleurs différentes. Il serait impensable pour un usager de changer à la main chaque couleur d'une image constituée de $256 \times 256 \times 256$ couleurs.

xv a aussi une interface spécialement conçue pour transformer des portions de couleurs pures vers une autre portion du spectre. La partie supérieure du centre de la figure 2.7 montre une portion de départ et sa portion finale. Ces régions ont été arbitrairement limitées à six régions dans *xv*. Chaque portion peut être transformée vers n'importe quel intervalle de couleurs monochromatiques indépendamment des autres portions de départ du spectre. Les intervalles finaux peuvent se superposer ou ne pas englober le spectre complet.

Une autre possibilité de transformation est le déplacement de la couleur considérée comme le blanc de référence. En déplaçant le blanc, on déplace toutes les couleurs de la gamut en maintenant leur position relative par rapport au blanc comme dans le milieu de la partie centrale de la figure 2.7. Toutes les couleurs de l'image sont modifiées par ce changement.

Toutes ces transformations ont un point en commun. Elles fonctionnent par essais et erreurs. L'utilisateur essaie une transformation avec tels paramètres, il regarde le résultat

et s'il n'est pas satisfait, il n'a qu'à essayer une autre combinaison de paramètres et de fonctions. Pour atteindre un résultat précis, l'utilisateur doit déterminer les sortes de transformations nécessaires, les paramètres de ces transformations ainsi que l'ordre d'application de ces transformations. Tout ça rend l'utilisation de ces logiciels assez complexe. L'utilisateur doit avoir beaucoup d'expérience pour ne pas naviguer en aveugle parmi toutes les combinaisons offertes.

La section suivante présente une autre branche de la transformation de couleurs : la transformation de gamut. La transformation de gamut fait partie des transformations de couleurs. Notre méthode serait donc applicable aux transformations de gamut, même si nous ne l'avons pas fait. Pour cette raison, nous présentons une section sur la transformation de gamut.

2.3 Transformation de gamut

La gamut d'un film, d'un moniteur ou d'une imprimante n'est pas la même. Selon la figure 2.8, il existe des couleurs affichables par un moniteur qui ne peuvent être reproduites par une imprimante et vice versa. La figure 2.8 présente un exemple de gamuts pour un film, un moniteur et une imprimante avec l'emplacement du "blanc" de chaque gamut. Dans cette figure, les gamuts sont représentées dans le diagramme de chromaticités CIE, mais il ne faut pas oublier qu'une gamut est un volume en trois dimensions dans un espace couleur. Pour reproduire sur film ou sur papier une image vue sur un moniteur, il faut transformer les couleurs qui ne pourront pas être reproduites afin de les inclure dans la gamut du médium désiré. On a le choix de transformer la gamut du moniteur pour l'adapter à la gamut de sortie, ou encore de transformer seulement la gamut de l'image à reproduire. La transformation de la gamut du moniteur procure l'uniformité de la transformation sur une série d'images, alors que la transformation de la gamut de l'image permet une reproduction adaptée aux spécificités de l'image. Dans un cas comme dans l'autre, la transformation de gamut est un problème en soi. Lorsque les couleurs d'une image se retrouvent à l'intérieur et à l'extérieur de la gamut de sortie, on doit faire un compromis entre reproduire identiquement les couleurs à l'intérieur des deux gamuts et transformer les couleurs pour maintenir les relations entre elles. Lorsqu'on reproduit identiquement les couleurs intérieures, les couleurs extérieures sont ramenées sur la frontière de la gamut et ainsi

on perd toutes les variations de ces couleurs. En faisant cela, on modifie l'apparence générale de l'image. L'oeil perçoit les couleurs en les comparant aux autres couleurs. Il ne perçoit pas la couleur absolue. Pour cette raison, il faut préconiser les différences relatives entre les couleurs plutôt que les différences absolues.

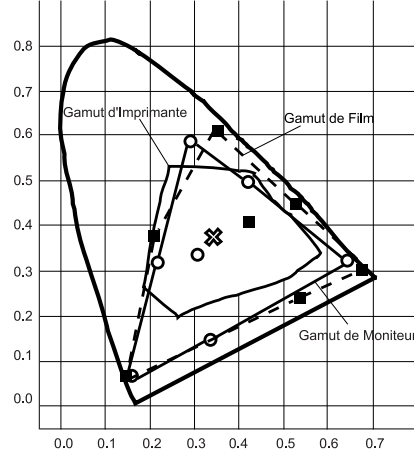


FIG. 2.8: Exemple de gamuts de moniteur, de film et d'imprimante. Le carré représente le "blanc" de la gamut du film. Le cercle représente celui du moniteur. La croix représente le "blanc" de la gamut de l'imprimante. Ce graphique provient de [SW91].

Stone et al. [SCB88] ont présenté une méthode pour faire la transformation de gamut. Cette méthode sert principalement pour adapter la gamut d'une image ou d'un moniteur à la gamut d'une imprimante. Ils ont établi des priorités à respecter pour préserver l'apparence d'une image. La priorité majeure préserve l'axe des gris de l'image. Ensuite, il faut préconiser le maximum de luminance pour obtenir le plus de contraste possible. Il faut aussi réduire le nombre de couleurs à l'extérieur de la gamut de l'imprimante et minimiser les changements de teinte et de saturation des couleurs. Finalement, il est préférable d'augmenter la saturation plutôt que de la diminuer. La transformation comme telle s'effectue dans l'espace CIE XYZ en utilisant l'équation suivante :

$$\begin{aligned} \mathbf{X}_d &= \mathbf{B}_d + bs * \mathbf{GA}_d + csf * \mathbf{R} * (\mathbf{X}_i - \mathbf{B}_i) \\ \mathbf{GA}_d &= \mathbf{W}_d - \mathbf{B}_d \end{aligned} \quad (2.2)$$

où les variables représentent :

\mathbf{B}_i	les coordonnées du noir de l'image,
\mathbf{B}_d	les coordonnées du noir de la gamut de sortie,
\mathbf{W}_d	les coordonnées du blanc de la gamut de sortie,
\mathbf{X}_i	les coordonnées de la couleur d'un pixel de l'image,
\mathbf{X}_d	les coordonnées de la couleur du pixel dans la gamut de sortie,
\mathbf{R}	une matrice de rotation,
\mathbf{GA}_d	l'axe des gris
bs	le facteur de translation du noir le long de \mathbf{GA}_d ,
csf	le facteur de changement d'échelle pour le contraste.

Premièrement, on applique une rotation à la gamut de l'image pour aligner les axes des gris des deux gamuts. Le paramètre csf est choisi pour que le blanc de l'image se retrouve juste à l'extérieur de la gamut de sortie pour avoir le maximum de contraste. Ensuite, on déplace la gamut du facteur de translation bs pour ne pas avoir trop de couleurs à l'extérieur de la gamut de sortie. L'augmentation de bs implique une diminution de csf . Il faut faire un compromis entre la saturation et le nombre de couleurs à l'extérieur de la gamut. L'augmentation de bs crée une désaturation de l'image tout en incluant plus de couleurs dans la gamut de sortie. L'ajustement final est laissé à l'appréciation de l'utilisateur.

Une autre façon de désaturer ou de sursaturer l'image est de déplacer les couleurs des phosphores utilisés : r , g et b . Si la gamut de sortie est plus petite que la gamut de l'image, on déplace les couleurs primaires vers le blanc. Si la somme des coordonnées des trois couleurs primaires reste la même qu'auparavant, les couleurs neutres vont le rester. Pour éviter un changement dans les teintes, il faut que les nouvelles couleurs soient sur les vecteurs allant du blanc aux anciennes couleurs. Cette règle est moins importante à respecter. Parfois, un léger changement de teinte compensé par des couleurs plus saturées est préférable. Finalement, les couleurs qui sont encore à l'extérieur de la gamut sont projetées perpendiculairement sur la face, l'arête ou le sommet le plus proche de la gamut de sortie.

Stone et Wallace [SW91] ont exposé une autre méthode pour faire la transformation de gamut. Cette fois-ci, ils travaillent dans l'espace CIE LAB qui est un espace perceptuellement plus uniforme et qui a des caractéristiques semblables à l'espace CIE LUV. Les principes qui les guident ont légèrement changé. Ils veulent surtout améliorer l'apparence générale de l'image et non reproduire exactement les couleurs de l'image lorsque la gamut de l'image n'est pas incluse dans la gamut de sortie. Leur première préoccupation est d'ajuster la luminance pour que les détails de l'image soient visibles

sur tout l'intervalle des tons. Il faut aussi ajuster les couleurs sans que l'appellation des couleurs change. Finalement, il faut maintenir ou légèrement augmenter la saturation relative de toute l'image.

La fonction permettant de passer de la gamut initiale à la gamut de sortie doit avoir certaines propriétés. Si la dérivée directionnelle de la fonction est presque nulle, la fonction réduit les détails de l'image puisqu'elle transforme plusieurs couleurs vers la même. Par contre, si la dérivée directionnelle est trop grande ou la fonction est discontinue, l'image sera quantifiée, c'est-à-dire que plusieurs couleurs seront transformées vers la même couleur. L'image finale aura moins de couleurs différentes que l'image originale. De plus, les composantes de la dérivée directionnelle doivent avoir le même signe que les composantes du vecteur de direction. Cette dernière restriction oblige les composantes couleurs de la gamut de sortie à augmenter lorsque les composantes de la gamut initiale augmentent.

Stone et Wallace traitent séparément la luminance et les composantes chromatiques. Pour la luminance, ils préconisent de faire les translations et les changements d'échelle tel que le blanc et le noir de la gamut de l'image se transforment en blanc et noir de la gamut de sortie. Cependant, le noir de l'image peut être transformé de façon que sa luminance soit un peu moindre que celle du noir de la gamut de sortie. En faisant cela, l'image finale aura plus de contraste, mais elle perdra des détails dans les régions sombres.

Pour les composantes chromatiques, ils compressent les couleurs le long du vecteur ayant une teinte constante afin d'éviter toute transformation entraînant un changement d'appellation dans les couleurs. La méthode la plus facile est de projeter les couleurs qui sont à l'extérieur de la gamut de sortie sur la frontière de la gamut la plus proche. Cette méthode maintient la saturation de l'image, mais elle crée de nouvelles taches sombres puisqu'elle transforme plusieurs couleurs vers une seule.

La seconde est d'utiliser une fonction linéaire dans la direction où la teinte est constante pour que toutes les couleurs soient comprises dans la gamut de sortie. Cette fonction ne crée pas des taches inexistantes, mais elle désature l'image. Les couleurs pastel pourraient devenir neutres. Stone et Wallace préfèrent utiliser une fonction qui a une tangente linéaire près de l'axe des gris et qui se rapproche de l'horizontale pour les couleurs saturées. La partie linéaire peut même avoir une pente plus grande que 1.

L'image finale n'en sera que plus saturée. Les trois fonctions sont présentées à la figure 2.9.

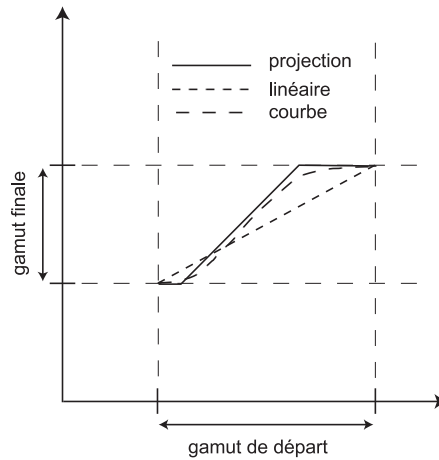


FIG. 2.9: Fonctions de compression : projection, linéaire et une courbe de Stone et Wallace [SW91].

La méthode de Stone et Wallace a parfois des problèmes. Ces problèmes sont dus au fait que la ligne, allant d'une couleur à la couleur neutre en maintenant la luminance constante, n'a pas une teinte constante. Une deuxième raison est que le blanc de chaque gamut n'a pas la même couleur. On peut toujours calculer les couleurs par rapport au blanc de la gamut, mais cela peut entraîner un changement dans l'appellation de certaines couleurs. À la toute fin, c'est à l'utilisateur de juger des résultats.

Power et al. [PWSS96] veulent améliorer la reproduction d'images imprimées en utilisant seulement deux encres, mais ces encres sont choisies en fonction de l'image à reproduire. Au lieu d'avoir trois ou quatre encres d'une couleur déterminée à l'avance, ils décrivent une procédure pour choisir les deux couleurs d'encre permettant la "meilleure" reproduction possible pour une image et un ensemble d'encres donnés. Pour définir la gamut produite par deux encres et une couleur de papier, ils utilisent le modèle de Neugebauer [Neu37]. Ils travaillent dans l'espace CIE XYZ. En connaissant la couleur exacte du papier, des deux encres séparées et superposées, ce modèle définit la gamut permise par une surface bilinéaire ayant pour sommets les quatre couleurs mentionnées.

La gamut d'impression n'étant pas la même que la gamut de l'image, il faut transformer la gamut de l'image pour l'inclure dans la gamut d'impression. Pour transformer la gamut de l'image qui est un volume en la gamut d'impression qui est une surface, ils utilisent un système d'axes orthogonaux. Ils transforment la gamut de l'image selon

deux dimensions et ils projettent parallèlement au troisième axe les couleurs sur la surface de la gamut d'impression. Les axes utilisés sont Y pour préserver la luminance relative, la direction de la plus grande variation de couleurs tout en restant orthogonal à Y et comme axe de projection : celui qui est orthogonal aux deux autres et proche de la normale à la surface de la gamut.

Ils transforment linéairement la luminance en premier pour que l'intervalle de luminance de l'image soit compris dans celui de la gamut d'impression. Pour chaque luminance, l'intervalle permis dans le sens du second axe n'est pas de la même grandeur. Ils discrétisent l'ensemble des couleurs en tranches perpendiculaires à la luminance. Pour chaque tranche, ils ramènent les couleurs dans l'intervalle permis selon trois méthodes : une transformation linéaire, une transformation qui projette perpendiculairement les couleurs extérieures sur la surface de la gamut et enfin une transformation selon une courbe de Bézier. Pour la courbe de Bézier, ils utilisent deux courbes de Bézier cubiques ayant une continuité C^1 . Cette courbe a une pente nulle aux extrémités de façon à compresser les couleurs extérieures à la gamut. Le milieu de la courbe est parallèle à la transformation linéaire afin de laisser intactes les couleurs au centre de la gamut. À cette étape, les couleurs sont toutes à l'intérieur de la gamut pour les deux premiers axes. Pour amener les couleurs sur la surface, ils projettent les couleurs sur la surface de la gamut parallèlement au troisième axe. Pour trouver les encres qui produiront un "meilleur" résultat selon l'image à reproduire, ils utilisent l'algorithme de recuit simulé (*simulated annealing*).

Stollnitz et al. [SOS98] ont poursuivi ces travaux pour permettre un nombre arbitraire d'encres. Le principe reste le même : l'utilisateur détermine combien d'encres qu'il veut utiliser parmi un ensemble d'encres donné. L'algorithme calcule la "meilleure" combinaison d'encres en fonction de l'image à reproduire. Ils étendent le modèle de Neugebauer à n encres en le combinant avec le modèle de Kubelka-Munk (décrit par Judd et Wyszecki [JW75] pp. 420–438) pour tenir compte de la réflexion de Fresnel. Avec ces deux modèles, ils peuvent déterminer la gamut d'une combinaison n d'encres et de papier. Si on utilise plus de deux encres, la gamut obtenue est un volume possiblement concave. Le problème de transformer la gamut de l'image vers la gamut d'impression est beaucoup plus complexe.

Certains [Lai87] [Mac93] [SCB88] utilisent une transformation cylindrique, c'est-à-

dire que la transformation se fait vers l'axe de la luminance en gardant la teinte et la luminance constantes. D'autres [Lai87] [Mac93] ont choisi une transformation sphérique : elle garde la teinte constante mais elle compresse les couleurs vers le centre de la gamut en changeant la luminance. Les auteurs ont préféré une méthode intermédiaire avec un paramètre permettant d'aller vers une ou l'autre selon l'image à reproduire. Si l'on utilise la méthode cylindrique, il faut transformer la luminance avant. Contrairement à l'article précédent [PWSS96], ils permettent à l'utilisateur de choisir entre trois fonctions pour la transformation : une fonction linéaire, une fonction qui projette les valeurs extérieures sur la surface et une fonction cubique qui compresse fortement les valeurs extrêmes et très peu les valeurs intermédiaires. Pour ce qui est de la transformation cylindrique, sphérique ou intermédiaire, l'utilisateur a encore le choix entre les trois fonctions précédentes. Pour choisir les encres, ils utilisent un algorithme génétique avec une probabilité qui diminue avec le temps.

Larson et al. [LRP97] présentent une méthode de compression pour l'affichage de scènes qui ont un intervalle de luminance plus grand que l'intervalle affichable. C'est une autre forme de transformation de gamut qui se fait principalement dans la dimension de la luminance. Leur méthode utilise l'histogramme de la luminance de l'image pour déterminer l'intervalle à compresser. Ils partent du principe que seule la luminance relative est importante pour l'observateur humain. Ils utilisent une égalisation de l'histogramme légèrement modifiée. Une égalisation normale de l'histogramme compresse les régions de l'histogramme qui ont peu de pixels et agrandit les régions ayant beaucoup de pixels. Si l'agrandissement est trop marqué, certaines régions ayant peu de changement de luminance produiront, tout à coup, des taches où la luminance est fortement modifiée sans avoir de correspondance réelle avec la scène. Pour éviter ce problème, ils obligent la pente de la fonction de transformation à ne pas dépasser la pente produite par une compression linéaire de la luminance. Avec cette approche, on obtient une image qui optimise la visibilité des détails dans les zones sombres comme dans les zones brillamment éclairées. Pour rendre les images plus conformes à la perception humaine, ils réajustent l'histogramme en fonction de la sensibilité humaine au contraste, de l'éblouissement produit par les zones plus brillantes, de la perte de la perceptibilité des couleurs et d'acuité visuelle dans un environnement sombre.

Dans ce chapitre, nous avons exposé les bases nécessaires à la compréhension des chapitres suivants. Nous avons présenté trois des espaces de couleurs les plus utilisés : l'espace CIE XYZ, l'espace RGB et l'espace perceptuellement plus uniforme CIE LUV. Nous avons discuté des possibilités des différents logiciels pour modifier les couleurs d'une image. Nous avons présenté un domaine connexe : la transformation de gamut. Ce domaine est très important dans la reproduction d'images. Ces transformations étant incluses dans les transformations de couleurs, certaines approches peuvent servir pour nos transformations.

Dans le prochain chapitre, nous aborderons comment évaluer les résultats obtenus après la transformation de couleurs. Il est important d'avoir un moyen autre que subjectif pour estimer si la transformation est "bonne" ou non.

Chapitre 3

Méthode d'évaluation des résultats

Avant de discuter des différentes transformations que nous pouvons faire, il faudrait avoir un moyen de comparer les résultats des diverses transformations entre elles. Il faut se donner une façon d'évaluer que telle transformation est “meilleure” que telle autre. Une première comparaison est purement subjective. Visuellement parlant, est-ce que l'utilisateur considère que la transformation donne une image plus proche du résultat escompté que la transformation précédente. Évidemment, lorsque les résultats sont semblables, il devient très difficile pour l'utilisateur d'ordonner les images de la meilleure à la pire. Il y a un autre problème. Étant donné que cette comparaison est subjective, le classement d'une série d'images effectué par un utilisateur peut être différent de celui fait par un second utilisateur. C'est pourquoi nous avons cherché une méthode plus objective de faire ces comparaisons.

3.1 RGB

Le résultat d'une transformation est une image aux couleurs semblables à l'image qu'on veut obtenir. Si les deux images sont faites à partir de scènes légèrement différentes, on peut comparer les couleurs des deux images pixel à pixel. Une première manière de comparer deux couleurs est de calculer la distance euclidienne en RGB entre les deux couleurs comme ceci :

$$DistRGB = \sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2} \quad (3.1)$$

où r_1, g_1 et b_1 sont les trois composantes RGB d'un pixel de la première image et r_2, g_2 et b_2 les composantes du pixel correspondant dans la seconde image. Dans une image, les composantes RGB d'un pixel sont des valeurs entières allant de 0 à 255. Lorsqu'on compare deux images, on est obligé d'utiliser les valeurs entières puisqu'on n'a pas d'autre information. Cependant, lorsqu'on transforme les couleurs d'une image, on connaît les valeurs réelles des composantes et on les utilise lors de la comparaison. Donc, si l'on compare une image originale avec une image transformée par nos soins, on utilise dans un cas des valeurs entières et dans l'autre des valeurs réelles pour les composantes couleurs.

En calculant la moyenne, la variance et la distance maximale sur tous les pixels, on obtient une méthode pour classifier les transformations. Les comparaisons peuvent se faire uniquement si les images de départ ont des scènes identiques. S'il y a trop de pixels où la scène est différente d'une image à l'autre, les distances calculées pour ces pixels vont considérablement augmenter la moyenne, la variance et la distance maximale au point qu'elles n'auront plus de signification.

3.2 LUV

Dans un premier temps, nous avons calculé la distance entre deux couleurs en RGB. Sachant que l'espace RGB n'est pas un espace perceptuellement uniforme, est-ce que la perception de la différence entre deux images correspond à la valeur de la distance en RGB ? En plus d'amplifier ou de diminuer la perception de certaines différences, le calcul en RGB peut classer une image comme étant plus proche de l'image voulue alors que perceptuellement c'est le contraire. La figure 3.1 en donne un exemple.

Supposons que l'image centrale de la figure 3.1 est le résultat qu'on veut obtenir. On veut comparer les deux autres images au résultat souhaité. La moyenne de la distance pixel à pixel entre l'image du haut et celle du centre en RGB est de 15.24 et de 7.18 en LUV. La distance moyenne entre l'image du bas et celle du centre est de 17.41 en RGB et de 5.98 en LUV. Selon le calcul en RGB, l'image la plus proche serait celle du haut, mais si on considère le calcul en LUV c'est le contraire. Une comparaison subjective confirme que l'image du bas est "meilleure" que la première.

Pour remédier à cela, nous transférons les couleurs de l'espace RGB qui sont en valeurs entières à l'espace LUV en valeurs réelles. Ensuite, nous calculons la distance

entre les deux couleurs dans cet espace. Pour passer de l'espace RGB à LUV [FvDFH90], il faut d'abord transformer les composantes RGB en CIE XYZ [SACM96] [Poy97] de cette façon :

$$\begin{aligned} X &= 0.412453 \times r + 0.35758 \times g + 0.180423 \times b \\ Y &= 0.212671 \times r + 0.71516 \times g + 0.072169 \times b \\ Z &= 0.019334 \times r + 0.119193 \times g + 0.950227 \times b. \end{aligned} \quad (3.2)$$

Ensuite, on calcule la luminance (L) de la manière suivante :

$$L = \begin{cases} 116 \sqrt[3]{Y} - 16 & \text{si } Y \geq 0.008856 \\ 903.3Y & \text{sinon.} \end{cases} \quad (3.3)$$

Et finalement, on obtient les composantes chromatiques (u et v) de cette façon :

$$\begin{aligned} u &= 13L \times \left(\frac{4X}{X + 15Y + 3Z} - 0.1978394 \right) \\ v &= 13L \times \left(\frac{9Y}{X + 15Y + 3Z} - 0.4683422 \right). \end{aligned} \quad (3.4)$$

La formule pour calculer la distance en LUV est la même qu'en RGB. On ne fait que changer les composantes r , g et b en L , u et v , respectivement.

$$DistLUV = \sqrt{(L_1 - L_2)^2 + (u_1 - u_2)^2 + (v_1 - v_2)^2}. \quad (3.5)$$

En calculant les mêmes statistiques que précédemment avec cette nouvelle fonction de distance, on a une méthode de comparaison qui est perceptuellement plus correcte. Les rouges et les bleus ne sont pas indûment surévalués par rapport aux verts.

3.3 L'importance de la variance

Même en calculant les statistiques avec la distance en LUV, on obtient parfois un classement qui ne satisfait pas notre perception visuelle. Dans la figure 3.2, on en retrouve un exemple.

L'image du haut dans la figure 3.2 a une moyenne en LUV de 4.19 et celle du bas de 4.39. La moyenne de l'image du haut est plus petite parce que la majorité des pixels ont une différence plus petite que dans celle du bas. Mais, il y a quelques pixels de l'image du haut qui sont très éloignés de ce qu'ils devraient être (31.4 en LUV). Ces pixels attirent le regard et sont beaucoup plus dérangeants qu'une différence moyenne un peu



FIG. 3.1: Défaillance du classement par la distance en RGB. Laquelle des deux images extrêmes est plus proche de celle au centre ?



FIG. 3.2: Défaillance du classement par la distance en LUV. Laquelle des deux images extrêmes est plus proche de celle au centre ?

plus grande mais uniforme sur toute une région. La variance est très importante pour distinguer les deux cas. La variance permet de savoir si beaucoup de pixels sont très éloignés de la moyenne. Les variances des deux images sont respectivement de 11.80 et de 7.00 et la différence maximale de 31.41 et de 20.69. La variance nous montre que les différences de couleur sont plus regroupées dans l'image du bas. D'après le maximum, on voit aussi que les valeurs marginales sont moins grandes que dans celle du haut.

La variance permet de voir si une image a plusieurs pixels ayant une grande différence de couleur ou si elle n'a que des différences mineures. Pour classer les images, il faudra toujours tenir compte de la moyenne et de la variance pour s'adapter aux particularités de perception de l'humain.

3.4 Calcul de l'erreur sur une partie de l'image

Mais toutes ces méthodes fonctionnent uniquement si les deux images comparées ont des scènes identiques. Lorsqu'elles sont trop différentes, les statistiques calculées avec n'importe quelle des trois fonctions n'ont que peu de signification.

Pour augmenter la classe des images qu'on peut comparer, on utilise ces statistiques mais calculées seulement sur une partie de l'image. L'utilisateur sélectionne une région d'intérêt de l'image et la comparaison se fait entre cette région et celle qui a le même emplacement dans la seconde image. Même si ce n'est pas une amélioration substantielle, elle permet de comparer des images qui ont un sous-ensemble de pixels où la scène est semblable. Comme exemple, prenons deux images du même point de vue où seul un objet a été enlevé de l'une des deux images. En calculant les statistiques sur toute l'image, la région contenant l'objet disparu créera des grandes différences d'erreur entre les deux images. En faisant les statistiques sur une région qui n'inclut pas cet objet, nous obtiendrons des données utilisables pour la classification des transformations. Malgré tout, ces méthodes de comparaison pixel à pixel ne permettent de comparer qu'une classe d'images. Si on prend une image et qu'on applique une translation d'un pixel pour créer la deuxième, les deux images seront considérées comme totalement différentes par les méthodes qu'on vient de discuter. La comparaison d'images pour déterminer à quel point elles sont semblables est un problème complexe comme l'ont constaté Rushmeier et al. [RWP⁺95], et les métriques n'en sont qu'au tout début.

Dans ce chapitre, nous avons vu deux méthodes pour comparer deux images : la distance dans l'espace RGB et la même distance dans l'espace LUV, sans oublier d'examiner la variance. Ces deux méthodes sont applicables sur les images entières ou des parties. Ceci nous donne en partie des moyens quantitatifs pour classer les transformations de couleurs entre elles, nous permettant d'évaluer laquelle est la "meilleure" et la "pire". Lorsque nécessaire et pour réduire les ambiguïtés, nous indiquerons aussi les jugements subjectifs de l'auteur. Dans le chapitre suivant, on abordera la méthode d'association par points faite par l'utilisateur et les diverses transformations réalisables à partir de ces associations.

Chapitre 4

Association des couleurs par l'utilisateur

Dans la réalité virtuelle, on utilise souvent des images panoramiques montrant ce qu'on voit dans toutes les directions [Che95]. Ces images panoramiques de 360° sont faites à partir de plusieurs photos. Ces photos sont prises dans suffisamment de directions pour qu'une partie de la scène se superpose d'une photo à l'autre. En principe, il suffit de superposer les photos pour avoir une image panoramique. En réalité, les couleurs d'un objet peuvent apparaître différemment d'une photo à l'autre. Ceci est dû à la combinaison de plusieurs facteurs tels l'inconstance de la caméra, du changement temporel de luminance entre la prise des photos, de la correction automatique des intensités de l'image par la caméra, du développement du négatif, etc. Si on superpose les photos sans faire de correction de couleurs, on risque d'observer une discontinuité marquée entre les photos.

Dans l'article de Pham et Pringle [PP95], les auteurs corrigent les couleurs d'une série de photos lorsqu'il y a superposition d'une photo à l'autre. Ils déterminent les parties des deux photographies qui se superposent et ils associent certaines couleurs de cette région de la première image à celles qui leur correspondent dans la seconde. Après, ils calculent la transformation qui permettra d'avoir les mêmes couleurs dans les parties superposées des deux images.

Cependant, l'article ne mentionne pas comment les couleurs sont choisies pour la correspondance. Le choix est-il fait par l'utilisateur ou automatiquement par le programme ? Dans un premier temps, il faut décider quelles couleurs sont caractéristiques de la

première image et ensuite les associer aux couleurs de la seconde image. Ces choix sont primordiaux puisque tout repose sur cette étape. On peut avoir le meilleur algorithme pour faire les transformations, mais si les couleurs sont mal associées, le résultat final sera tout de même désastreux.

Dans ce chapitre, on verra une première méthode pour faire ces correspondances. On présentera aussi les différentes transformations qu'on a explorées et qui utilisent ces correspondances.

4.1 Méthode d'association

La méthode la plus simple pour associer des couleurs d'une image à l'autre est de demander à l'utilisateur de le faire. L'utilisateur clique sur un pixel dans la première image et le met en correspondance avec un pixel de la seconde image en cliquant dessus. Le programme associe la couleur des deux pixels ensemble. L'utilisateur peut sélectionner le nombre de paires qu'il veut. La figure 4.1 donne un exemple de cinq associations. Chaque point correspond au point ayant le même numéro dans l'autre image.

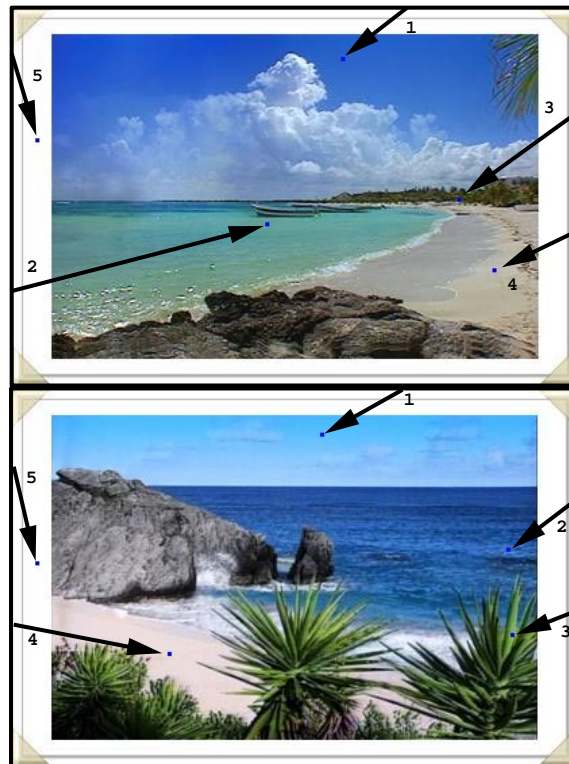


FIG. 4.1: Mise en correspondance de couleurs par l'utilisateur.

Les avantages de cette méthode de correspondance sont la simplicité de l'algorithme et la rapidité d'exécution. Ce qui ralentit le processus est le temps que met l'utilisateur à choisir les pixels qu'il veut mettre en correspondance. La sélection repose sur le jugement de l'utilisateur, ce qui lui donne un contrôle plus grand qu'une sélection automatique. Mais s'il choisit mal ses correspondances, le résultat en souffrira.

4.2 Transformations dans l'espace RGB

Une fois qu'on a associé les couleurs d'une image avec celles d'une autre image, il y a plusieurs types de transformations qu'on peut utiliser afin que les couleurs transformées soient le plus près possible des couleurs voulues.

Il existe un autre domaine où l'on utilise des associations pour faire des transformations. Ce domaine est le recalage d'objets, c'est-à-dire la transformation d'un objet pour qu'il ressemble à un modèle en fait de grandeur et d'alignement afin de pouvoir les comparer. Pour faire cette transformation, plusieurs types de transformations ont été explorées. Subsol a travaillé avec des transformations rigides, affines, polynomiales et avec des splines [Sub95].

Les transformations rigides permettent de faire des rotations et des translations. Pour une transformation rigide en trois dimensions, on a besoin d'une matrice de rotation R :

$$R = \begin{pmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \\ c_1 & c_2 & c_3 \end{pmatrix} \quad (4.1)$$

et d'un vecteur de translation (a_4, b_4, c_4) . En combinant les deux ensembles, on obtient le système suivant :

$$\begin{aligned} x' &= a_1x + a_2y + a_3z + a_4 \\ y' &= b_1x + b_2y + b_3z + b_4 \\ z' &= c_1x + c_2y + c_3z + c_4 \end{aligned} \quad (4.2)$$

où

$$R^t R = I \text{ et } \det(R) = 1.$$

Ce type de transformation n'a pas beaucoup de souplesse. Dans le cas des transformations de couleurs, cette rigidité n'est pas nécessaire et ne s'adapte pas aux contraintes des associations.

Pour les transformations affines, on utilise le même système que celui de l'équation 4.2 sauf qu'il n'y a plus de contraintes sur la matrice de rotation R . Ces transformations comprennent toutes les rotations, les translations, les changements d'échelle, les cisaillements et les combinaisons des quatre précédents. Pour les transformations de couleurs, ce type de transformations est intéressant puisqu'il combine bon nombre de transformations présentes dans les logiciels existants.

Les transformations polynomiales comprennent toutes les transformations pouvant s'exprimer par des fonctions polynomiales. Le système suivant représente la transformation d'ordre deux ou quadratique :

$$\begin{aligned} x' &= a_1x^2 + a_2y^2 + a_3z^2 + a_4xy + a_5xz + a_6yz + a_7x + a_8y + a_9z + a_{10} \\ y' &= b_1x^2 + b_2y^2 + b_3z^2 + b_4xy + b_5xz + b_6yz + b_7x + b_8y + b_9z + b_{10} \\ z' &= c_1x^2 + c_2y^2 + c_3z^2 + c_4xy + c_5xz + c_6yz + c_7x + c_8y + c_9z + c_{10}. \end{aligned} \quad (4.3)$$

Ces transformations d'ordre plus élevé permettent plus de liberté à la transformation. Cependant, il faut faire attention parce qu'une transformation d'ordre trop élevé a tendance à osciller et les résultats en souffriront. Il est préférable d'exécuter des transformations quadratiques un certain nombre de fois plutôt que de prendre une transformation d'ordre trop élevé.

Les transformations avec des splines utilisent des fonctions B-splines ou de Bézier. La rigidité de ces fonctions procure une certaine régularité dans la transformation. Une autre caractéristique intéressante est qu'une association a une influence locale. Donc, les associations mal choisies n'auront pas d'influence sur toutes les couleurs, leur influence est limitée à une région restreinte.

Pham et Pringle [PP95] corrigent des séries de photos ayant une partie superposée à la suivante par des associations de couleurs et des transformations. Ils déterminent les régions superposées des deux photos au pixel près. Pour choisir les associations, ils prennent un certain nombre de pixels dans la région superposée de la première photo et associent ces couleurs aux pixels qui leur correspondent dans la région superposée de l'autre photo. La transformation modifie toutes les couleurs de la photo, pas uniquement les couleurs de la région superposée. En prenant les associations seulement dans cette région, les couleurs du reste de la photo peuvent être transformées vers un résultat insatisfaisant. Pour éviter ce problème, ils ajoutent un certain nombre de couleurs provenant du reste de la photo et ils associent la couleur à elle-même. Plus le

rapport entre ces associations et celles entre les régions superposées est grand plus la photo restera la même après la transformation. Pour les transformations, ils utilisent des transformations cubiques dans différents espaces de couleurs en itérant ces transformations jusqu'à l'obtention d'un résultat satisfaisant. Ce résultat est jugé seulement de façon subjective.

Les types de transformations explorées pour ce mémoire sont polynomiales dans l'espace RGB dans cette section et dans l'espace LUV pour la prochaine section.

4.2.1 Linéaire

Pour une transformation linéaire ordinaire, il faut trouver les coefficients du système d'équations suivant :

$$\begin{aligned} r_2 &= T_{r1}r_1 + T_{r2}g_1 + T_{r3}b_1 + T_{r4} \\ g_2 &= T_{g1}r_1 + T_{g2}g_1 + T_{g3}b_1 + T_{g4} \\ b_2 &= T_{b1}r_1 + T_{b2}g_1 + T_{b3}b_1 + T_{b4} \end{aligned} \tag{4.4}$$

où r_1, g_1 et b_1 sont les composantes RGB de l'image de départ et r_2, g_2 et b_2 sont les composantes de l'image qu'on veut obtenir. Les T_i sont les coefficients qu'il faut calculer pour faire une transformation linéaire.

Puisque ce système a trois équations et 12 inconnues, il faut associer quatre couleurs linéairement indépendantes de la première image aux couleurs provenant de la deuxième image pour pouvoir résoudre ce système. Étant donné que les couleurs sont encodées en valeurs entières (0 à 255 pour chaque composante), il est pratiquement impossible d'obtenir une solution exacte même si les inconnues sont calculées en valeurs réelles. Pour avoir une "meilleure" approximation, il faut travailler avec plus de correspondances. Avec la technique des moindres carrés, on peut associer autant de couleurs que l'on veut et cette méthode trouvera la transformation qui est la plus "proche" de toutes ces contraintes.

Pour arriver au système d'équations de la méthode des moindres carrés, on doit faire la somme des carrés de chaque ligne du système 4.4 en remplaçant les composantes RGB par les couleurs des pixels choisis. Les associations choisies sont p_{11}, \dots, p_{1n} pour les pixels de la première image et p_{21}, \dots, p_{2n} pour la seconde. La couleur du pixel p_{ij}

est représentée par (r_{ij}, g_{ij}, b_{ij}) . Le système 4.4 devient :

$$\begin{aligned} S_{r_2} &= \sum_{i=1}^n (r_{2i} - T_{r1}r_{1i} - T_{r2}g_{1i} - T_{r3}b_{1i} - T_{r4})^2 \\ S_{g_2} &= \sum_{i=1}^n (g_{2i} - T_{g1}r_{1i} - T_{g2}g_{1i} - T_{g3}b_{1i} - T_{g4})^2 \\ S_{b_2} &= \sum_{i=1}^n (b_{2i} - T_{b1}r_{1i} - T_{b2}g_{1i} - T_{b3}b_{1i} - T_{b4})^2. \end{aligned} \quad (4.5)$$

La solution qu'on recherche est la valeur de chaque coefficient T_i qui minimise ces sommes (S). Pour ce faire, on doit dériver chaque équation du système 4.5 par rapport à chaque coefficient inconnu. Pour la première ligne du système 4.5, on obtient les équations suivantes :

$$\begin{aligned} \frac{\partial S_{r_2}}{\partial T_{r1}} &= \sum_{i=1}^n 2(r_{2i} - T_{r1}r_{1i} - T_{r2}g_{1i} - T_{r3}b_{1i} - T_{r4})(-r_{1i}) \\ \frac{\partial S_{r_2}}{\partial T_{r2}} &= \sum_{i=1}^n 2(r_{2i} - T_{r1}r_{1i} - T_{r2}g_{1i} - T_{r3}b_{1i} - T_{r4})(-g_{1i}) \\ \frac{\partial S_{r_2}}{\partial T_{r3}} &= \sum_{i=1}^n 2(r_{2i} - T_{r1}r_{1i} - T_{r2}g_{1i} - T_{r3}b_{1i} - T_{r4})(-b_{1i}) \\ \frac{\partial S_{r_2}}{\partial T_{r4}} &= \sum_{i=1}^n 2(r_{2i} - T_{r1}r_{1i} - T_{r2}g_{1i} - T_{r3}b_{1i} - T_{r4})(-1). \end{aligned} \quad (4.6)$$

Les autres lignes se dérivent de la même manière. On obtient 12 équations. Les valeurs des coefficients qui minimisent le système d'équations 4.4 rendent les équations précédentes nulles. Pour la transformation des rouges, on a le système suivant à résoudre :

$$\begin{aligned} \sum_{i=1}^n r_{2i}r_{1i} &= T_{r1} \sum_{i=1}^n r_{1i}^2 + T_{r2} \sum_{i=1}^n r_{1i}g_{1i} + T_{r3} \sum_{i=1}^n r_{1i}b_{1i} + T_{r4} \sum_{i=1}^n r_{1i} \\ \sum_{i=1}^n r_{2i}g_{1i} &= T_{r1} \sum_{i=1}^n r_{1i}g_{1i} + T_{r2} \sum_{i=1}^n g_{1i}^2 + T_{r3} \sum_{i=1}^n g_{1i}b_{1i} + T_{r4} \sum_{i=1}^n g_{1i} \\ \sum_{i=1}^n r_{2i}b_{1i} &= T_{r1} \sum_{i=1}^n r_{1i}b_{1i} + T_{r2} \sum_{i=1}^n g_{1i}b_{1i} + T_{r3} \sum_{i=1}^n b_{1i}^2 + T_{r4} \sum_{i=1}^n b_{1i} \\ \sum_{i=1}^n r_{2i} &= T_{r1} \sum_{i=1}^n r_{1i} + T_{r2} \sum_{i=1}^n g_{1i} + T_{r3} \sum_{i=1}^n b_{1i} + T_{r4}n. \end{aligned} \quad (4.7)$$

La méthode des moindres carrés linéaire donne dans notre situation trois systèmes d'équations qu'on résoud par l'élimination de Gauss avec pivotage ([BF93] algorithmes

6.2 et 6.3). Ces systèmes sont résumés dans la matrice augmentée ci-dessous :

$$\left(\begin{array}{cccc|c} T_{r1} \sum_{i=1}^n r_{1i}^2 & T_{r2} \sum_{i=1}^n r_{1i} g_{1i} & T_{r3} \sum_{i=1}^n r_{1i} b_{1i} & T_{r4} \sum_{i=1}^n r_{1i} & \sum_{i=1}^n r_{2i} r_{1i} \\ T_{r1} \sum_{i=1}^n r_{1i} g_{1i} & T_{r2} \sum_{i=1}^n g_{1i}^2 & T_{r3} \sum_{i=1}^n g_{1i} b_{1i} & T_{r4} \sum_{i=1}^n g_{1i} & \sum_{i=1}^n r_{2i} g_{1i} \\ T_{r1} \sum_{i=1}^n r_{1i} b_{1i} & T_{r2} \sum_{i=1}^n g_{1i} b_{1i} & T_{r3} \sum_{i=1}^n b_{1i}^2 & T_{r4} \sum_{i=1}^n b_{1i} & \sum_{i=1}^n r_{2i} b_{1i} \\ T_{r1} \sum_{i=1}^n r_{1i} & T_{r2} \sum_{i=1}^n g_{1i} & T_{r3} \sum_{i=1}^n b_{1i} & T_{r4} n & \sum_{i=1}^n r_{2i} \end{array} \right). \quad (4.8)$$

La matrice 4.8 permet de trouver les coefficients affectant la composante rouge. Pour la composante verte, il suffit de substituer les T_{gi} aux T_{ri} et les g_{2i} aux r_{2i} . Pour la composante bleue, c'est la même chose : T_{bi} et b_{2i} remplacent T_{ri} et r_{2i} respectivement.

Une fois qu'on a calculé les coefficients inconnus T_{ri} , T_{gi} et T_{bi} , on se sert de ces valeurs dans l'équation 4.4. On transforme chaque pixel de l'image de départ en l'insérant dans cette équation (4.4).

Résultats

Dans un premier temps, on a vérifié que la transformation linéaire calculée par la méthode des moindres carrés retrouvait bien les transformations linéaires. Pour ce faire, on a pris les couleurs d'une image et on leur a appliqué une rotation en RGB pour créer la seconde image. Ensuite, nous avons expérimenté la transformation sur ces deux images.

Dans la figure 4.2, l'image du haut est celle de départ. L'image à gauche est l'image obtenue en appliquant sur les couleurs la rotation et la translation suivante :

$$\begin{pmatrix} r_2 \\ g_2 \\ b_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} r_1 \\ g_1 \\ b_1 \\ 1 \end{pmatrix}. \quad (4.9)$$

Les couleurs obtenues après cette transformation sont arrondies pour obtenir des valeurs entières. Pour faciliter la discussion, nous mentionnerons cet exemple sous le nom de *rotation*. Pour créer l'image de droite, l'utilisateur a choisi 10 correspondances en prenant soin de sélectionner des couleurs différentes les unes des autres et représentatives de la distribution des couleurs dans l'image. Comme l'image de gauche représente la même scène que celle de départ, on fait correspondre chaque pixel choisi dans la première image à celui qui a les mêmes coordonnées dans la seconde. L'erreur moyenne entre l'image de gauche et celle de droite en LUV est de 0.178 et la variance de 0.0779. Les

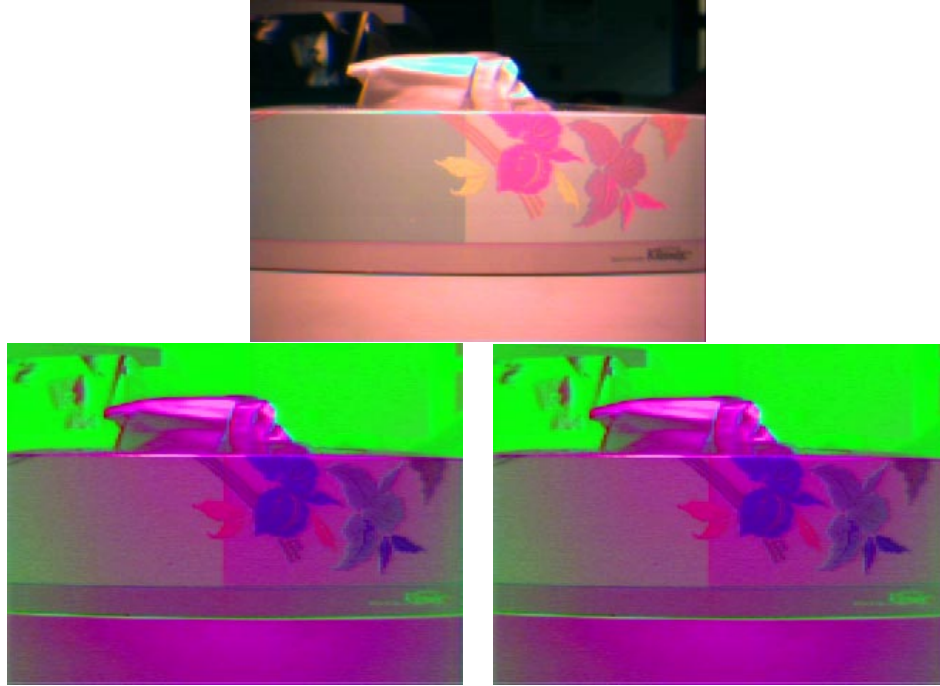


FIG. 4.2: Résultats d'une transformation linéaire pour retrouver une rotation et une translation. En haut : l'image de départ. En bas à gauche : le résultat qu'on veut obtenir. En bas à droite : le résultat obtenu. Les images ont 320 par 240 pixels.

erreurs sont calculées en prenant la valeur réelle des pixels sans la limiter au cube RGB après la transformation de l'image. Bien entendu, il faut ramener ces valeurs réelles à des entiers pour l'affichage. Les coefficients de transformation trouvés par le programme sont :

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ -1.00034 & -0.00279 & 0.00245 & 1.00003 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \quad (4.10)$$

Si, au lieu de laisser l'utilisateur décider quelles couleurs sélectionner, le programme les choisit à l'aide de pseudo-hasard, les résultats sont un peu moins bons mais tout aussi acceptables. Encore une fois, les couleurs correspondantes dans la seconde image sont celles des pixels ayant les mêmes positions que ceux de la première. Le tableau 4.1 résume les résultats. Dans le tableau 4.1, les colonnes *Erreur moy LUV*, *Variance de l'erreur LUV*, *Erreur max LUV* et *Différence max coeff.* représentent respectivement l'erreur moyenne en LUV, la variance de l'erreur en LUV, l'erreur maximale en LUV et la différence la plus grande entre les coefficients de transformation et ceux de la rotation réelle.

Comme second résultat (figure 4.3), nous avons transformé la même image que dans

Image et nombre de correspondances	Erreur moy LUV	Variance de l'erreur LUV	Erreur max LUV	Différence max coeff.
haut et gauche	78.7	1620	161	—
gauche et droite 10 corres. choisies	0.178	0.0779	1.20	0.003
gauche et droite 10 corres. hasard	0.337	0.214	6.12	0.04
gauche et droite 8 corres. hasard	0.362	0.263	11.07	0.05

TAB. 4.1: Résultats de la transformation linéaire en RGB pour une rotation avec translation.

la figure 4.2 avec cette fois-ci la combinaison d'un cisaillement et d'un changement d'échelle. Les coefficients de la transformation donnant l'image de gauche de la figure 4.3 sont :

$$\begin{pmatrix} 0.5 & 0 & 0.5 & 0 \\ 0 & 0.6 & 0.4 & 0 \\ 0 & 0 & 0.7 & 0 \end{pmatrix}. \quad (4.11)$$

Cet exemple sera appelé ultérieurement *cisaillement*. L'image de droite est obtenue par notre algorithme en prenant les mêmes 10 associations choisies par l'utilisateur. L'erreur moyenne en LUV est de 0.369 et la variance de 0.0926 en LUV. Les coefficients calculés par le programme sont :

$$\begin{pmatrix} 0.4990 & -0.00648 & 0.50578 & -0.000102 \\ -0.00116 & 0.600279 & 0.400022 & -0.000781 \\ -0.00148 & 0.00368 & 0.6969 & -0.000911 \end{pmatrix}. \quad (4.12)$$

Le tableau 4.2 donne les mêmes renseignements que le tableau 4.1. Évidemment, l'erreur varie en fonction des points qui sont choisis. Si les couleurs choisies sont très distinctes les unes des autres et représentatives de la distribution des couleurs dans l'image, on obtiendra de bons résultats. Pour montrer l'influence de la sélection des points, on a recommencé 50 fois la sélection de 10 points au hasard pour les images des figures 4.2 et 4.3. Le tableau 4.3 donne la moyenne des 50 essais, ainsi que la variance et le maximum de l'erreur moyenne en LUV pour les images transformées linéairement. Comme on peut le constater, l'influence du choix des correspondances dans ces deux exemples n'est pas énorme. Même l'ensemble de points qui produit l'erreur moyenne maximale donne une image visuellement difficile à différencier de l'objectif. Pour retrouver une transformation linéaire, la méthode des moindres carrés linéaire avec 10

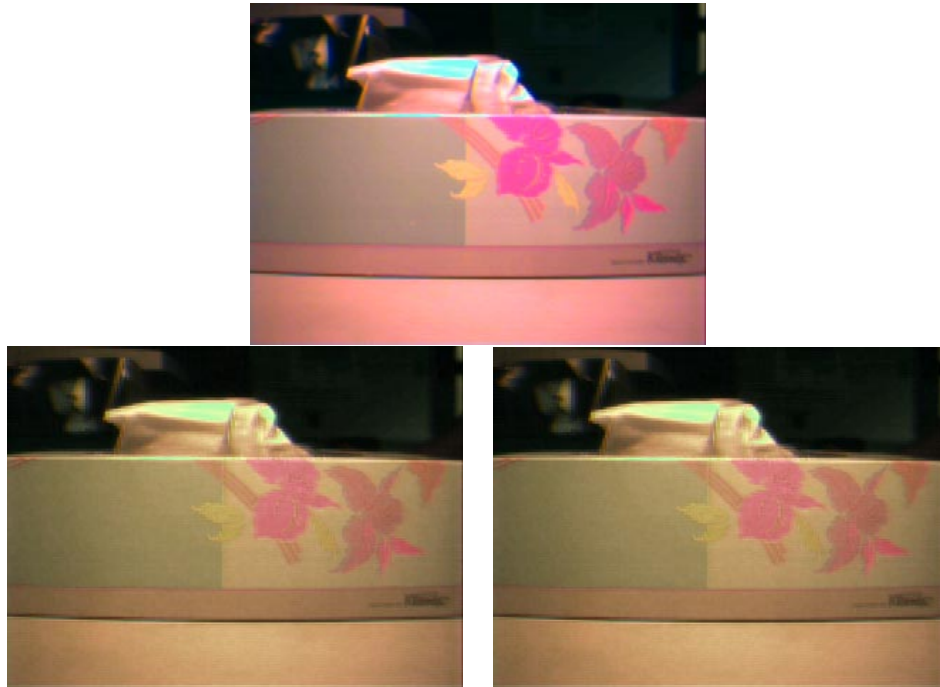


FIG. 4.3: Résultats d'une transformation linéaire pour retrouver la combinaison d'un cisaillement et d'un changement d'échelle. En haut : l'image de départ. En bas à gauche : le résultat qu'on veut obtenir. En bas à droite : le résultat obtenu. Les images ont 320 par 240 pixels.

correspondances donne un résultat acceptable si les couleurs choisies sont bien distribuées. Dans la section 4.4, nous verrons l'effet du nombre de correspondances sur l'erreur moyenne.

Nous obtenons donc les résultats voulus lorsqu'il existe une transformation linéaire permettant de passer de l'image de départ à l'image définissant l'objectif. Cependant, il n'existe pas toujours une transformation linéaire reliant les images. Ceci nous conduit à utiliser la méthode des moindres carrés sur les équations quadratiques.

4.2.2 Quadratique

Pour déduire les systèmes d'équations utilisant la méthode des moindres carrés pour des fonctions quadratiques, on part du système d'équations pour une transformation quadratique :

$$\begin{aligned}
r_2 &= T_{r1}r_1^2 + T_{r2}g_1^2 + T_{r3}b_1^2 + T_{r4}r_1g_1 + T_{r5}r_1b_1 + T_{r6}g_1b_1 + T_{r7}r_1 + T_{r8}g_1 + T_{r9}b_1 + T_{r10} \\
g_2 &= T_{g1}r_1^2 + T_{g2}g_1^2 + T_{g3}b_1^2 + T_{g4}r_1g_1 + T_{g5}r_1b_1 + T_{g6}g_1b_1 + T_{g7}r_1 + T_{g8}g_1 + T_{g9}b_1 + T_{g10} \\
b_2 &= T_{b1}r_1^2 + T_{b2}g_1^2 + T_{b3}b_1^2 + T_{b4}r_1g_1 + T_{b5}r_1b_1 + T_{b6}g_1b_1 + T_{b7}r_1 + T_{b8}g_1 + T_{b9}b_1 + T_{b10}
\end{aligned}
\tag{4.13}$$

où r_1, g_1 et b_1 sont les composantes couleurs en RGB de l'image de départ et r_2, g_2 et b_2 sont les composantes qu'on veut obtenir. Les T_{ri}, T_{gi} et T_{bi} sont les coefficients à calculer pour une transformation quadratique.

Image et nombre de correspondances	Erreur moy LUV	Variance de l'erreur LUV	Erreur max LUV	Différence max coeff.
haut et gauche	18.7	48.6	90.6	—
gauche et droite 10 corres. choisies	0.369	0.0926	2.68	0.006
gauche et droite 10 corres. hasard	0.519	0.163	4.27	0.04
gauche et droite 8 corres. hasard	0.519	0.175	4.45	0.04

TAB. 4.2: Résultats de la transformation linéaire en RGB pour un cisaillement avec changement d'échelle.

Figures	Moyenne	Variance	Maximum
fig. 4.2 : <i>rotation</i>	0.270	0.00492	0.493
fig. 4.3 : <i>cisaillement</i>	0.503	0.0223	1.03

TAB. 4.3: L'influence de la sélection des points pour les fig. 4.2 : *rotation* et 4.3 : *cisaillement* avec une transformation linéaire. Statistiques sur l'erreur moyenne en LUV pour 50 essais avec 10 correspondances prises au hasard.

Cette fois pour résoudre le système 4.13, il nous faut 10 correspondances linéairement indépendantes entre les couleurs de l'image de départ et celles de l'objectif. Pour obtenir le système d'équations de la méthode des moindres carrés pour les fonctions quadratiques, on procède de la même façon que dans la section linéaire. Les

systèmes résultant de ce processus sont donnés dans le système matriciel suivant :

$$\begin{pmatrix}
 r_{1_i}^4 & r_{1_i}^2 g_{1_i}^2 & r_{1_i}^2 b_{1_i}^2 & r_{1_i}^3 g_{1_i} & r_{1_i}^3 b_{1_i} & r_{1_i}^2 g_{1_i} b_{1_i} & r_{1_i}^3 & r_{1_i}^2 g_{1_i} & r_{1_i}^2 b_{1_i} & r_{1_i}^2 \\
 r_{1_i}^2 g_{1_i}^2 & g_{1_i}^4 & g_{1_i}^2 b_{1_i}^2 & r_{1_i} g_{1_i}^3 & r_{1_i} g_{1_i}^2 b_{1_i} & g_{1_i}^3 b_{1_i} & r_{1_i} g_{1_i}^2 & g_{1_i}^3 & g_{1_i}^2 b_{1_i} & g_{1_i}^2 \\
 r_{1_i}^2 b_{1_i}^2 & g_{1_i}^2 b_{1_i}^2 & b_{1_i}^4 & r_{1_i} g_{1_i} b_{1_i}^2 & r_{1_i} b_{1_i}^3 & g_{1_i} b_{1_i}^3 & r_{1_i} b_{1_i}^2 & g_{1_i} b_{1_i}^2 & b_{1_i}^3 & b_{1_i}^2 \\
 r_{1_i}^3 g_{1_i} & r_{1_i} g_{1_i}^3 & r_{1_i} g_{1_i} b_{1_i}^2 & r_{1_i}^2 g_{1_i}^2 & r_{1_i}^2 g_{1_i} b_{1_i} & r_{1_i} g_{1_i}^2 b_{1_i} & r_{1_i}^2 g_{1_i} & r_{1_i} g_{1_i}^2 & r_{1_i} g_{1_i} b_{1_i} & r_{1_i} g_{1_i} \\
 r_{1_i}^3 b_{1_i} & r_{1_i} g_{1_i}^2 b_{1_i} & r_{1_i} b_{1_i}^3 & r_{1_i}^2 g_{1_i} b_{1_i} & r_{1_i}^2 b_{1_i}^2 & r_{1_i} g_{1_i} b_{1_i}^2 & r_{1_i}^2 b_{1_i} & r_{1_i} g_{1_i} b_{1_i} & r_{1_i} b_{1_i}^2 & r_{1_i} b_{1_i} \\
 r_{1_i}^2 g_{1_i} b_{1_i} & g_{1_i}^3 b_{1_i} & g_{1_i} b_{1_i}^3 & r_{1_i} g_{1_i}^2 b_{1_i} & r_{1_i} g_{1_i} b_{1_i}^2 & g_{1_i}^2 b_{1_i}^2 & r_{1_i} g_{1_i} b_{1_i} & g_{1_i}^2 b_{1_i} & g_{1_i} b_{1_i}^2 & g_{1_i} b_{1_i} \\
 r_{1_i}^3 & r_{1_i} g_{1_i}^2 & r_{1_i} b_{1_i}^2 & r_{1_i}^2 g_{1_i} & r_{1_i}^2 b_{1_i} & r_{1_i} g_{1_i} b_{1_i} & r_{1_i}^2 & r_{1_i} g_{1_i} & r_{1_i} b_{1_i} & r_{1_i} \\
 r_{1_i}^2 g_{1_i} & g_{1_i}^3 & g_{1_i} b_{1_i}^2 & r_{1_i} g_{1_i}^2 & r_{1_i} g_{1_i} b_{1_i} & g_{1_i}^2 b_{1_i} & r_{1_i} g_{1_i} & g_{1_i}^2 & g_{1_i} b_{1_i} & g_{1_i} \\
 r_{1_i}^2 b_{1_i} & g_{1_i}^2 b_{1_i} & b_{1_i}^3 & r_{1_i} g_{1_i} b_{1_i} & r_{1_i} b_{1_i}^2 & g_{1_i} b_{1_i}^2 & r_{1_i} b_{1_i} & g_{1_i} b_{1_i} & b_{1_i}^2 & b_{1_i} \\
 r_{1_i}^2 & g_{1_i}^2 & b_{1_i}^2 & r_{1_i} g_{1_i} & r_{1_i} b_{1_i} & g_{1_i} b_{1_i} & r_{1_i} & g_{1_i} & b_{1_i} & n
 \end{pmatrix}
 \times
 \begin{pmatrix}
 T_{r1} \\
 T_{r2} \\
 T_{r3} \\
 T_{r4} \\
 T_{r5} \\
 T_{r6} \\
 T_{r7} \\
 T_{r8} \\
 T_{r9} \\
 T_{r10}
 \end{pmatrix}
 =
 \begin{pmatrix}
 r_{2_i} r_{1_i}^2 \\
 r_{2_i} g_{1_i}^2 \\
 r_{2_i} b_{1_i}^2 \\
 r_{2_i} r_{1_i} g_{1_i} \\
 r_{2_i} r_{1_i} b_{1_i} \\
 r_{2_i} g_{1_i} b_{1_i} \\
 r_{2_i} r_{1_i} \\
 r_{2_i} g_{1_i} \\
 r_{2_i} b_{1_i} \\
 r_{2_i}
 \end{pmatrix}
 \quad (4.14)$$

Pour afficher ici la taille de ce système, on a enlevé les signes de sommations. Tous les éléments de la matrice de gauche et de celle de droite sont des sommations $\sum_{i=1}^n$. Ce système permet de calculer la transformation pour la composante rouge. Pour les deux autres composantes, la matrice de gauche reste la même. Dans la matrice des coefficients, on remplace les T_{ri} par T_{gi} ou T_{bi} selon la composante voulue. Pour la partie droite de l'équation 4.14, on substitue g_{2_i} ou b_{2_i} à r_{2_i} selon la composante souhaitée. On solutionne ces trois systèmes par l'élimination de Gauss avec pivotage pour trouver tous les coefficients.

Résultats

Pour vérifier la précision de la méthode des moindres carrés quadratique, on a repris les mêmes images de départ que celles de la figure 4.2 : *rotation* en gardant les mêmes correspondances. La transformation de couleur entre l'image de départ et celle illustrant l'objectif est une rotation suivie d'une translation. Cette transformation étant linéaire, la transformation quadratique doit obtenir des résultats semblables à la méthode précédente. La figure 4.4 montre l'application de la transformation quadratique pour retrouver une rotation.

L'erreur moyenne en LUV entre les deux images de la figure 4.4 est de 1.29 et la variance de 2.33 comparativement à 0.178 en LUV et la variance de 0.0779 pour la transformation linéaire. L'erreur moyenne est plus élevée lorsqu'on utilise une trans-

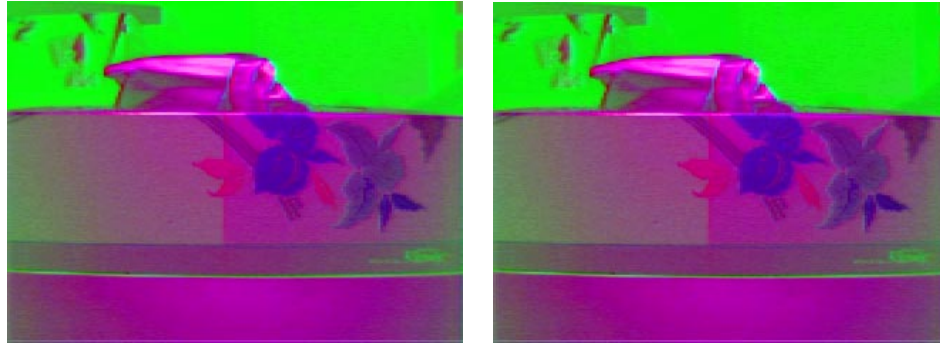


FIG. 4.4: Résultats d'une transformation quadratique sur les images de la figure 4.2 : *rotation* en utilisant les mêmes correspondances. À gauche : l'image objectif. À droite : le résultat obtenu.

formation quadratique plutôt qu'une transformation linéaire pour retrouver une transformation linéaire parce que celle-là demande plus de correspondances pour stabiliser les résultats. Il faut un minimum de 10 correspondances pour que la transformation quadratique puisse être stable. Si les couleurs sont mal choisies, alors plus de couleurs peuvent être nécessaires pour stabiliser les résultats. Le tableau 4.4 montre les erreurs obtenues en utilisant 10 correspondances choisies par l'utilisateur, et 10 et 15 correspondances prises au hasard.

Image et nombre de correspondances	Erreur moy LUV	Variance de l'erreur LUV	Erreur max LUV	Différence max coeff.
haut et gauche fig. 4.2	78.7	1620	161	—
fig. 4.4; 10 corres. choisies	1.29	2.33	25.8	0.6
fig. 4.4; 10 corres. hasard	2.81	55.69	162.6	4
fig. 4.4; 15 corres. hasard	0.932	4.15	65.5	1

TAB. 4.4: Résultats de la transformation quadratique en RGB pour une rotation avec translation.

On reprend également l'exemple du cisaillement suivi d'un changement d'échelle. La figure 4.5 est le résultat de la transformation quadratique en se servant de l'image de départ et de celle objective de la figure 4.3 avec les mêmes 10 correspondances. L'erreur moyenne entre les images de la figure 4.5 est de 0.979 en LUV et la variance est de



FIG. 4.5: Résultats d'une transformation quadratique sur les images de la figure 4.3 : *cisaillement* en utilisant les mêmes correspondances. À gauche : l'image objectif. À droite : le résultat obtenu.

1.64, comparativement à 0.369 pour la moyenne en LUV et la variance de 0.0926 pour la transformation linéaire. Le tableau 4.5 montre les différentes mesures d'erreur pour 10 correspondances choisies par l'utilisateur versus 10 et 15 correspondances prises au hasard.

Image et nombre de correspondances	Erreur moy LUV	Variance de l'erreur LUV	Erreur max LUV	Différence max coeff.
haut et gauche fig. 4.3	18.7	48.6	90.6	—
fig. 4.5 ; 10 corres. choisies	0.979	1.64	25.6	0.2
fig. 4.5 ; 10 corres. hasard	2.08	16.68	75.0	2
fig. 4.5 ; 15 corres. hasard	1.29	2.90	25.0	0.4

TAB. 4.5: Résultats de la transformation quadratique en RGB pour un cisaillement avec changement d'échelle.

Lorsque la transformation qu'on veut trouver est linéaire, la transformation quadratique prend beaucoup plus de correspondances pour obtenir des résultats semblables à ceux de la transformation linéaire. À la section 4.4, on discutera du nombre de correspondances par rapport à l'erreur moyenne. Par contre, la transformation quadratique permet de retrouver une plus grande classe de transformations. Un exemple d'une transformation non-linéaire est donné à la figure 4.6. Le cadre et les coins font partie des images. Cet exemple sera appelé *gamma*.



FIG. 4.6: Résultats d'une transformation quadratique pour retrouver une transformation gamma. En haut : l'image de départ. En bas à gauche : l'image objectif. En bas à droite : le résultat obtenu. Les images ont 420 par 280 pixels.

L'image de départ est celle du haut de la figure 4.6. L'image de gauche a été créée en appliquant une transformation gamma de 2.5 pour chaque dimension de RGB sur l'image du haut. Le système d'équations qui a permis de créer l'image objectif est le suivant :

$$\begin{aligned} r_2 &= r_1^{2.5} \\ g_2 &= g_1^{2.5} \\ b_2 &= b_1^{2.5} \end{aligned} \tag{4.15}$$

où r_1, g_1 et b_1 sont les composantes RGB de l'image de départ et r_2, g_2 et b_2 sont celles de l'image objectif. Cette transformation n'est même pas quadratique. L'image de droite est le résultat d'une transformation quadratique faite à l'aide de 15 correspondances. L'erreur moyenne en LUV, la variance ainsi que le maximum de l'erreur en LUV sont donnés au tableau 4.6 pour 15 et 25 correspondances. Puisque la transformation gamma n'est pas quadratique, il faut plus de points pour espérer bien cerner la transformation.

Le tableau 4.7 montre l'influence du choix des correspondances pour les images des figures 4.2 : *rotation*, 4.3 : *cisaillement* et 4.6 : *gamma* pour la transformation quadratique. Pour chaque figure, on a appliqué la transformation quadratique sur 50 ensembles

Image et nombre de correspondances	Erreur moy LUV	Variance de l'erreur LUV	Erreur max LUV
haut et gauche	26.2	237	60.7
gauche et droite; 15 corres. hasard	9.49	432.2	109.7
gauche et droite; 25 corres. hasard	2.12	15.8	40.3

TAB. 4.6: Résultats de la transformation quadratique en RGB pour transformation gamma de 2.5.

de correspondances. Le tableau présente la moyenne, la variance et le maximum de l'erreur moyenne obtenus par ces ensembles.

Figure	Nombre de points	Moyenne	Variance	Maximum
fig. 4.2 : rotation	10 points	5.46	76.96	53.21
	15 points	0.888	0.353	3.64
fig. 4.3 : cisaillement	10 points	6.86	51.64	36.61
	15 points	1.17	0.491	5.07
fig. 4.6 : gamma	15 points	3.91	12.74	20.49
	25 points	1.99	1.07	8.32

TAB. 4.7: L'influence de la sélection des points pour les fig. 4.2 : *rotation*, 4.3 : *cisaillement* et 4.6 : *gamma* avec une transformation quadratique. Statistiques sur l'erreur moyenne en LUV pour 50 essais avec des correspondances prises au hasard.

D'après le tableau 4.7, on voit que pour retrouver une transformation linéaire, il faut au moins 15 correspondances à la méthode des moindres carrés quadratique. Avec la méthode linéaire, il faut moins de points (10 est très suffisant) pour avoir de meilleurs résultats. La transformation quadratique est avantageuse lorsque la transformation entre les deux images est non-linéaire. On peut même avec plus de correspondances (environ 30) ajuster assez bien une transformation non-quadratique : la transformation gamma 2.5 de la figure 4.6.

Lorsque la transformation entre les images de départ est due à un changement de luminance ou d'éclairage, une transformation dans l'espace LUV serait peut-être plus appropriée puisque cet espace est plus perceptuel. La prochaine section est consacrée aux résultats de l'exploration des transformations de couleurs dans l'espace LUV.

4.3 Transformations dans l'espace LUV

Pour toutes les transformations de couleurs dans l'espace LUV, il faut d'abord encoder les couleurs choisies en LUV. Pour ce faire, on utilise les équations 3.2 à 3.4 vues au chapitre 3. On applique la transformation sur les couleurs en LUV. Ensuite, on ramène les couleurs modifiées dans l'espace RGB et on projette les couleurs qui sont à l'extérieur du cube RGB sur la face du cube la plus proche. On calcule les coefficients des transformations avec la méthode des moindres carrés. La première transformation applique la méthode des moindres carrés linéaire sur les trois composantes : L , u et v . Les deux dernières méthodes de transformations traitent séparément la luminance et les composantes chromatiques. Cette idée de séparer la luminance du reste vient de plusieurs articles sur la transformation de gamuts [SCB88] [SW91] [PWSS96]. Ils commencent par ajuster la luminance pour qu'elle reste dans l'intervalle permis par la gamut en préservant la luminance relative. Ensuite, ils modifient la chromaticité. La deuxième transformation utilise la méthode des moindres carrés linéaire sur la luminance et un autre moindres carrés linéaire sur u et v . La dernière transformation traite la luminance de la même façon que la précédente mais elle utilise un moindres carrés cubique pour u et v . Nous utiliserons des parenthèses pour indiquer les composantes que la transformation traite ensemble.

4.3.1 Linéaire en (Luv)

La transformation linéaire en (Luv) se dérive de la même façon que la transformation linéaire en RGB. Le système d'équations 4.16 est identique au système 4.4 sauf que L , u et v remplacent respectivement r , g et b .

$$\begin{aligned}
 L_2 &= T_{L1}L_1 + T_{L2}u_1 + T_{L3}v_1 + T_{L4} \\
 u_2 &= T_{u1}L_1 + T_{u2}u_1 + T_{u3}v_1 + T_{u4} \\
 v_2 &= T_{v1}L_1 + T_{v2}u_1 + T_{v3}v_1 + T_{v4}
 \end{aligned} \tag{4.16}$$

où L_1, u_1 et v_1 sont les composantes LUV de l'image de départ, L_2, u_2 et v_2 sont celles de l'image objectif et les T_{Li}, T_{ui} et T_{vi} sont les coefficients de la transformation qu'il faut calculer. Pour passer à la forme des moindres carrés, les équations illustrant les différentes opérations sont identiques aux équations 4.5 à 4.8 en remplaçant r, g et b par L, u et v . On résout le système obtenu par l'élimination de Gauss avec pivotage.

Résultats

Cette nouvelle transformation de couleurs devrait être plus adaptée aux changements mineurs d'illumination. Pour vérifier ceci, on utilise deux photos prises par une caméra numérique inconstante sous un éclairage identique. L'éclairage principal est fourni par une ampoule bleue. L'image de départ qu'on veut corriger est légèrement rosée par rapport à la seconde image. La seconde sera utilisée comme image objectif. La figure 4.7 montre le résultat d'une transformation linéaire en (Luv) avec les 20 associations prises au hasard mais hors de la zone du vase et de son ombre. Puisque les images ne proviennent pas de scènes identiques, on calcule l'erreur seulement sur une partie des images. Cette région rectangulaire représente la droite de l'image et est illustrée par la figure 4.8. L'erreur moyenne en LUV sur cette région est de 2.46 et la variance de 2.70. Cet exemple sera appelé *inconstance de la caméra*.

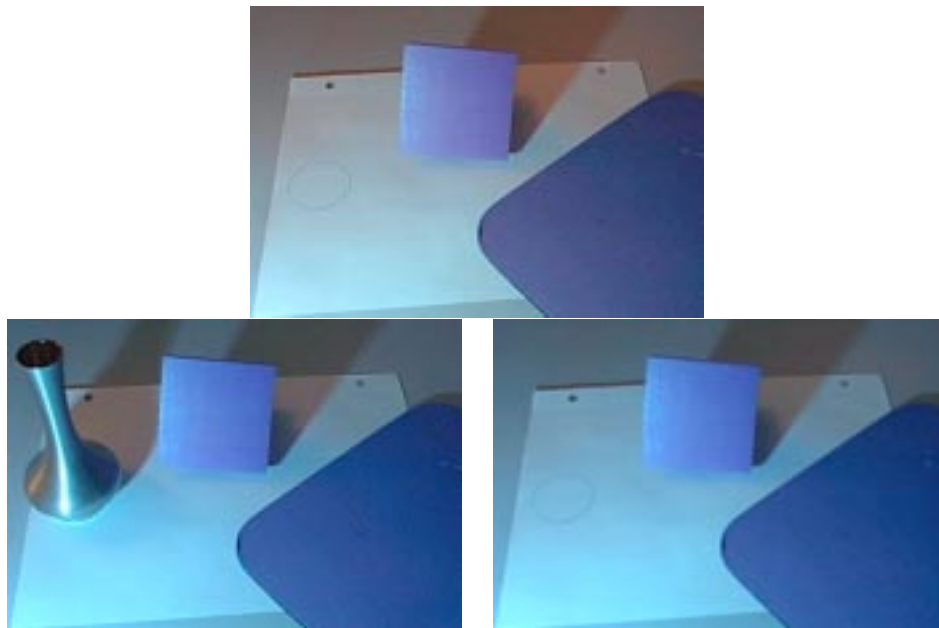


FIG. 4.7: Résultats d'une transformation linéaire en (Luv) pour retrouver l'*inconstance de la caméra*. En haut : l'image de départ. En bas à gauche : l'image objectif. En bas à droite : le résultat obtenu. Les images ont 380 par 260 pixels.

L'exemple de l'*inconstance de la caméra* se prête assez bien à une transformation de couleurs en l'espace LUV. Par contre, cette transformation donne de très mauvais résultats lorsqu'elle correspond à une rotation du cube RGB entre les deux images. La figure 4.9 donne le résultat de la transformation avec 50 correspondances pour

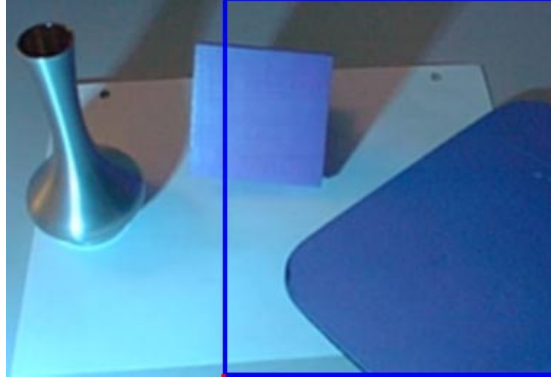


FIG. 4.8: Région où est calculée l'erreur pour la fig. 4.7 : *inconstance de la caméra*.

retrouver la *rotation* de la figure 4.2. Malgré le nombre de correspondances, l'erreur moyenne entre les pixels en LUV est de 19.4 et la variance de 163, comparativement à 0.178 en moyenne et 0.0779 pour la variance avec une transformation linéaire en RGB et seulement 10 correspondances. La rotation en RGB qui est une transformation linéaire en RGB devient une transformation non-linéaire en LUV. Une transformation linéaire en RGB aurait autant de difficultés à retrouver une rotation en LUV.

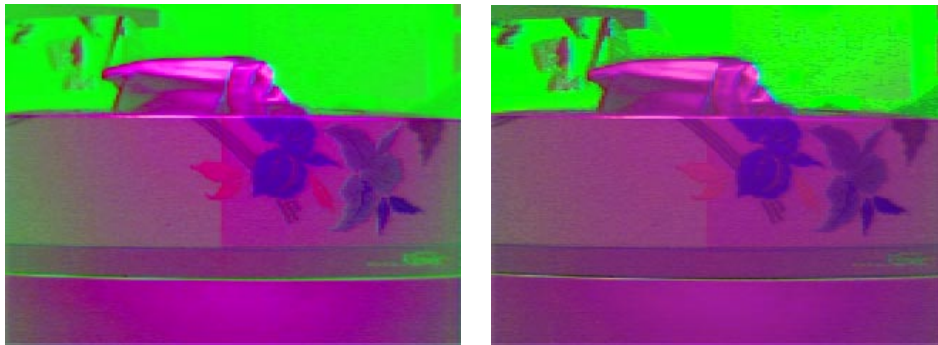


FIG. 4.9: Résultats d'une transformation linéaire en (*Luv*) sur les images de la figure 4.2 : *rotation* en utilisant 50 correspondances.

Le tableau 4.8 présente les variations de l'erreur moyenne en LUV selon la sélection des points. On a choisi 50 fois au hasard des ensembles de correspondances. Le tableau donne la moyenne, la variance et le maximum de l'erreur moyenne entre les pixels des images objectif et résultantes en LUV. Pour les images de *l'inconstance de la caméra*, les correspondances sont prises au hasard hors de la zone contenant le vase et son ombre puisqu'ils ne sont pas présents dans les deux images de départ. L'erreur est calculée seulement dans la région indiquée à la figure 4.8.

Figure	Nombre de points	Moyenne	Variance	Maximum
fig. 4.2 : <i>rotation</i>	50 points	19.22	0.973	21.99
fig. 4.7 : <i>inconstance de la caméra</i>	20 points	2.99	0.127	4.32
	50 points	2.69	0.0222	3.19

TAB. 4.8: L'influence de la sélection des points pour les fig. 4.2 : *rotation* et 4.7 : *inconstance de la caméra* avec une transformation linéaire en (Luv) . Statistiques sur l'erreur moyenne en LUV pour 50 essais avec des correspondances prises au hasard. Les correspondances pour la fig. 4.7 sont prises dans toute l'image sauf dans la zone du vase et de son ombre.

Comme nous l'avons indiqué précédemment, nous examinerons si la séparation de la luminance et des composantes chromatiques peut faciliter la transformation des couleurs comme elle le fait pour la transformation de gamut.

4.3.2 Linéaire en (L) et (uv)

Cette méthode traite séparément la luminance L et les composantes chromatiques u et v . On utilise l'équation suivante :

$$L_2 = T_{L1}L_1 + T_{L2} \quad (4.17)$$

pour dériver la méthode des moindres carrés linéaire pour la luminance. Dans l'équation 4.17, L_1 et L_2 sont les composantes de luminance dans l'image de départ et dans l'image objectif. Les T_{Li} sont les coefficients qu'il faut trouver. En passant par les étapes illustrées par les équations 4.5 à 4.7, on obtient le système matriciel suivant :

$$\begin{pmatrix} \sum_{i=1}^n L_{2i}L_{1i} \\ \sum_{i=1}^n L_{2i} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n L_{1i}^2 & \sum_{i=1}^n L_{1i} \\ \sum_{i=1}^n L_{1i} & n \end{pmatrix} \begin{pmatrix} T_{L1} \\ T_{L2} \end{pmatrix} \quad (4.18)$$

où n est le nombre d'associations, L_{1i} est la composante de luminance du i^e pixel choisi de l'image de départ et L_{2i} de l'image objectif.

Pour les composantes chromatiques, on part du système d'équations linéaire suivant :

$$\begin{aligned} u_2 &= T_{u1}u_1 + T_{u2}v_1 + T_{u3} \\ v_2 &= T_{v1}u_1 + T_{v2}v_1 + T_{v3} \end{aligned} \quad (4.19)$$

où u_1 et v_1 sont les composantes chromatiques de l'image de départ et u_2 et v_2 celles de l'image objectif. Les T_{ui} et T_{vi} sont les coefficients de la transformation qu'il faut calculer.

En appliquant la méthode des moindres carrés sur le système précédant, on obtient les deux systèmes matriciels suivants :

$$\begin{pmatrix} \sum_{i=1}^n u_{2_i} u_{1_i} \\ \sum_{i=1}^n u_{2_i} v_{1_i} \\ \sum_{i=1}^n u_{2_i} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n u_{1_i}^2 & \sum_{i=1}^n u_{1_i} v_{1_i} & \sum_{i=1}^n u_{1_i} \\ \sum_{i=1}^n u_{1_i} v_{1_i} & \sum_{i=1}^n v_{1_i}^2 & \sum_{i=1}^n v_{1_i} \\ \sum_{i=1}^n u_{1_i} & \sum_{i=1}^n v_{1_i} & n \end{pmatrix} \begin{pmatrix} T_{u1} \\ T_{u2} \\ T_{u3} \end{pmatrix} \quad (4.20)$$

$$\begin{pmatrix} \sum_{i=1}^n v_{2_i} u_{1_i} \\ \sum_{i=1}^n v_{2_i} v_{1_i} \\ \sum_{i=1}^n v_{2_i} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n u_{1_i}^2 & \sum_{i=1}^n u_{1_i} v_{1_i} & \sum_{i=1}^n u_{1_i} \\ \sum_{i=1}^n u_{1_i} v_{1_i} & \sum_{i=1}^n v_{1_i}^2 & \sum_{i=1}^n v_{1_i} \\ \sum_{i=1}^n u_{1_i} & \sum_{i=1}^n v_{1_i} & n \end{pmatrix} \begin{pmatrix} T_{v1} \\ T_{v2} \\ T_{v3} \end{pmatrix}$$

où n est le nombre d'associations, u_{1_i} et v_{1_i} sont les composantes chromatiques du i^e pixel de l'image de départ et u_{2_i} et v_{2_i} sont celles du i^e pixel de l'image objectif. Les systèmes 4.18 et 4.20 sont résolus par l'élimination de Gauss avec pivotage. Le nombre d'inconnues dans les équations 4.17 et 4.19 impose un minimum de trois associations linéairement indépendantes avant que les résultats de la méthode se stabilisent.

Résultats

Pour comparer les résultats avec ceux obtenus par la transformation précédente, nous reprenons l'exemple de l'*inconstance de la caméra*. La figure 4.10 montre l'image objectif et le résultat de la transformation linéaire en (L) et (uv) avec les 20 correspondances utilisées pour la figure 4.7. L'erreur moyenne en Luv sur la région désignée à la figure 4.8 est de 2.73 et la variance de 2.79 comparativement à 2.46 et 2.70 pour la transformation linéaire en (Luv) .



FIG. 4.10: Résultats d'une transformation linéaire en (L) et (uv) pour retrouver l'*inconstance de la caméra*. À gauche : l'image objectif. À droite : le résultat obtenu.

On voit que la transformation linéaire en (L) et en (uv) est un peu moins bonne que celle qui est linéaire en (Luv) pour l'exemple de l'*inconstance de la caméra*. Mais ce n'est rien comparé au résultat désastreux obtenu sur les images de l'exemple de la *rotation*. La figure 4.10 présente le résultat avec 50 correspondances ainsi que l'image objectif. L'erreur moyenne en LUV est de 39.8 et la variance est de 895. Pour la transformation linéaire en (Luv) , l'erreur moyenne était de 19.4 en LUV et la variance de 163.

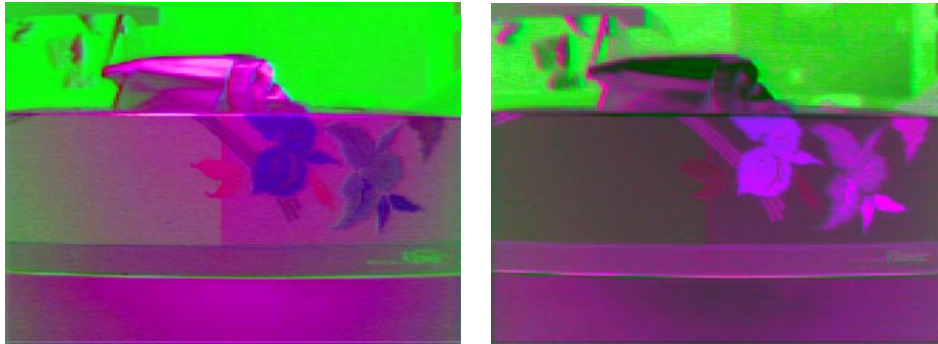


FIG. 4.11: Résultats d'une transformation linéaire en (L) et (uv) sur les images de la figure 4.2 : *rotation* en utilisant 50 correspondances.

Pourquoi une rotation des couleurs en RGB est-elle si imprécise à traiter dans l'espace LUV ? En séparant la luminance des composantes chromatiques, on ne peut retrouver avec succès que les transformations qui gardent la monotonie de la luminance. Ce qui veut dire que les couleurs de l'image de départ classées en ordre croissant de luminance doivent, une fois transformées, former une fonction monotone de luminance. Une rotation du cube RGB ne maintient pas la monotonie de la luminance. Donc, toute tentative de correspondance linéaire entre les luminances est vouée à l'échec. Évidemment, l'angle de rotation a de l'importance. Si l'angle est suffisamment petit, la luminance ne sera pas assez affectée pour donner des résultats inacceptables. Pour déterminer la grandeur des angles permis, on a calculé la luminance des huit sommets du cube RGB après une rotation selon un axe du cube. On place les huit sommets en ordre croissant de luminance avant la rotation. Si, après la rotation, la luminance des sommets n'est plus monotone, alors l'angle est trop grand. Cette méthode ne donne qu'une approximation de l'angle maximal permis selon les trois axes de rotation puisqu'un angle plus grand pourrait être admis selon les couleurs apparaissant dans les images. De plus ces angles varient selon l'axe de rotation car une rotation selon l'axe des gris ne pose aucun problème quelque soit l'angle. Le tableau 4.9 donne l'approximation des

angles selon les axes R, G et B.

Axe de rotation	Angle max. positif	Angle max. négatif
Rouge	5 degrés	11 degrés
Vert	26 degrés	18 degrés
Bleu	24 degrés	11 degrés

TAB. 4.9: Approximation des angles maximaux de rotation selon les axes R, G et B.

Pour bien montrer que les résultats obtenus aux figures 4.10 et 4.11 ne sont pas dus à la sélection des points, on a choisi au hasard 50 fois le nombre de correspondances voulues. Le tableau 4.10 donne la moyenne, la variance et le maximum de l'erreur moyenne en LUV obtenus par la transformation linéaire en (L) et (uv) de ces ensembles d'associations.

Figure	Nombre de points	Moyenne	Variance	Maximum
fig. 4.7 : <i>inconstance de la caméra</i>	20 points	3.15	0.0758	3.94
	50 points	3.01	0.0346	3.54
fig. 4.2 : <i>rotation</i>	50 points	39.58	6.74	47.51

TAB. 4.10: L'influence de la sélection des points pour les fig. 4.2 : *rotation* et 4.7 : *inconstance de la caméra* avec une transformation linéaire en (L) et (uv) . Statistiques sur l'erreur moyenne en Luv pour 50 essais avec des correspondances prises au hasard. Les correspondances pour la fig. 4.7 sont prises dans toute l'image sauf dans la zone du vase et de son ombre.

Les résultats obtenus à la figure 4.10 ne sont pas aussi bons qu'on pourrait s'y attendre. Pour voir d'où provient la plus grande partie de l'erreur, on a calculé la différence moyenne entre la luminance des images et la distance euclidienne dans les composantes chromatiques. La distance entre les composantes chromatiques est nettement plus grande que pour la luminance. C'est pourquoi, nous avons pensé qu'une augmentation du degré des équations chromatiques serait appropriée.

4.3.3 Linéaire en (L) et cubique en (uv)

Pour cette transformation, on traite la luminance de la même manière que dans la transformation linéaire en (L) et (uv) . Pour le calcul de la luminance, on utilise l'équation 4.17. Pour la partie chromatique, on applique la méthode des moindres carrés cubique. On part des équations du troisième degré pour u et v

$$\begin{aligned}
u_2 &= T_{u1}u_1^3 + T_{u2}v_1^3 + T_{u3}u_1^2v_1 + T_{u4}u_1v_1^2 + T_{u5}u_1^2 + T_{u6}v_1^2 + T_{u7}u_1v_1 \\
&\quad + T_{u8}u_1 + T_{u9}v_1 + T_{u10} \\
v_2 &= T_{v1}u_1^3 + T_{v2}v_1^3 + T_{v3}u_1^2v_1 + T_{v4}u_1v_1^2 + T_{v5}u_1^2 + T_{v6}v_1^2 + T_{v7}u_1v_1 \\
&\quad + T_{v8}u_1 + T_{v9}v_1 + T_{v10}
\end{aligned} \tag{4.21}$$

où u_1 et v_1 sont les composantes chromatiques de l'image de départ, u_2 et v_2 sont celles de l'image objectif et les T_{ij} sont les coefficients de la transformation qu'il faut trouver. Pour utiliser ces équations dans le cadre de la méthode des moindres carrés, on refait les mêmes opérations qui nous ont permis d'obtenir les équations 4.5 à 4.7. Pour calculer les inconnues, on utilise l'élimination de Gauss avec pivotage. Avec les équations du troisième degré pour les composantes chromatiques, il faut un minimum de 10 correspondances où u et v sont linéairement indépendants pour atteindre la stabilité des résultats.

Résultats

Pour comparer les résultats de cette transformation, on reprend l'exemple de l'*inconstance de la caméra* de la figure 4.7. La figure 4.12 montre le résultat d'une transformation linéaire en (L) et cubique en (uv) avec les mêmes 20 correspondances de la figure 4.7. L'erreur moyenne entre l'image objectif et l'image résultat de la figure 4.12 calculée seulement sur la région désignée est de 3.73 en LUV et 10.7 pour la variance. Le résultat, loin de s'améliorer, s'est dégradé en utilisant la méthode des moindres carrés cubique pour u et v . Ceci est dû à la difficulté de stabiliser les résultats. Dans ces images, les composantes chromatiques des pixels sont suffisamment dépendantes les unes des autres. Les 10 associations de couleurs indépendantes nécessaires à la stabilisation de la partie cubique sont assez difficiles à trouver.

Le tableau 4.11 donne l'erreur moyenne en LUV et la variance de l'erreur en LUV pour les cinq transformations discutées précédemment pour 20 et 50 correspondances. Comme le montre le tableau, on n'obtient pas toujours de "meilleurs" résultats en prenant plus de points. Les transformations linéaires en RGB et en (Luv) donnent des résultats presque équivalents. Les autres transformations produisent des images plus éloignées de celle voulue.



FIG. 4.12: Résultat d'une transformation linéaire en (L) et cubique en (uv) sur les images de la figure 4.7 : *inconstance de la caméra* en utilisant les mêmes correspondances. À gauche : l'image objectif. À droite : le résultat obtenu.

Nombre de points	Linéaire en RGB		Quadratique en RGB		Linéaire en (Luv)		Linéaire en $(L)(uv)$		Linéaire en (L) cubique en (uv)	
	moy	var	moy	var	moy	var	moy	var	moy	var
20 points	2.42	2.72	3.02	4.94	2.46	2.70	2.73	2.79	3.73	10.66
50 points	2.68	2.52	2.76	2.87	2.64	2.46	2.96	2.55	3.10	3.62

TAB. 4.11: Résultats des différentes transformations pour l'*inconstance de la caméra* avec 20 et 50 correspondances.

Pour montrer que ces 20 correspondances sont bien représentatives, on a sélectionné 50 fois au hasard des ensembles de 20 et 50 points. Les statistiques de cette opération sont présentées dans le tableau 4.12. Le tableau donne la moyenne, la variance et le maximum de l'erreur moyenne en LUV de ces 50 essais pour la région choisie précédemment.

Figure	Nombre de points	Moyenne	Variance	Maximum
fig. 4.7 : caméra	20 points	6.44	16.7	27.39
	50 points	3.22	0.0967	4.36

TAB. 4.12: L'influence de la sélection des points pour la fig. 4.7 : *inconstance de la caméra* avec une transformation linéaire en (L) et cubique en (uv) . Statistiques sur l'erreur moyenne pour une région en LUV pour 50 essais avec des correspondances prises au hasard hors de la zone du vase et de son ombre.

La prochaine section fait une comparaison plus poussée des cinq transformations de couleurs étudiées.

4.4 Résultats comparatifs des transformations précédentes

Pour bien comparer les résultats des différentes transformations, on utilise six exemples : la *rotation*, le *cisaillement*, le *gamma*, l'*inconstance de la caméra* et deux nouveaux exemples. Ces deux derniers sont présentés à la figure 4.13. La transformation de couleurs permettant de passer de l'image en haut à gauche à celle d'en haut à droite est une rotation en RGB avec une compression de certaines couleurs et elle affecte tous les pixels de la même façon. On appellera cet exemple la *compression*. Les images du bas représentent une scène prise par une caméra sous un éclairage différent. L'une des photos est éclairée par une ampoule blanche de 100 watts tandis que l'autre l'est par une ampoule rouge de 60 watts. La position des sources lumineuses est similaire mais pas identique. Ceci causera des erreurs sur le contour des ombres. Cet exemple sera nommé le *changement d'éclairage*.

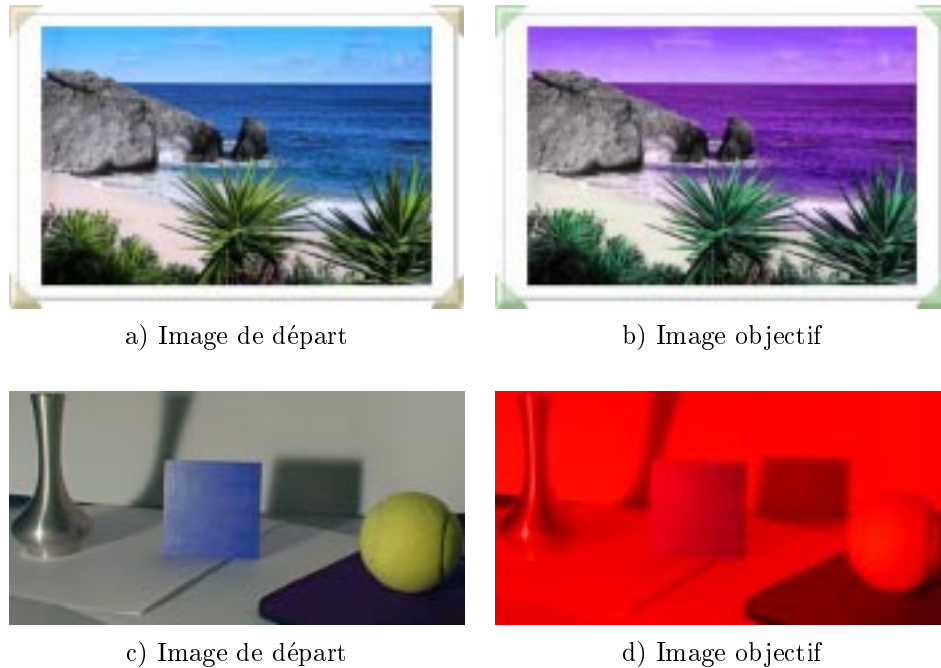


FIG. 4.13: Deux nouveaux exemples. En haut : une rotation et une compression de couleurs. Ces images ont 420 par 280 pixels. En bas : un changement d'éclairage. Ces images ont 388 par 200 pixels.

Pour le temps d'exécution, toutes les transformations se font de façon interactive. Lorsque l'utilisateur a fini de sélectionner une paire de couleurs pour une nouvelle associa-

tion, la transformation est calculée et affichée avant que l'utilisateur ait choisi une nouvelle paire. Cette interaction continue même avec 50 associations. En fait, l'augmentation du nombre d'associations n'influence pas beaucoup le temps de calcul. Il est plus coûteux de transformer chaque pixel de l'image de départ avec la transformation calculée que de trouver les coefficients de la transformation à l'aide d'un moindres carrés. Pour cette raison, les transformations dans l'espace LUV sont un peu moins performantes que dans l'espace RGB tout en restant interactives. Pour les transformations dans l'espace LUV, on doit transformer tous les pixels de l'image dans l'espace LUV, exécuter la transformation calculée et retransformer ces couleurs dans l'espace RGB pour l'affichage.

Pour chacun des six exemples, on montre l'influence du nombre de correspondances choisies sur l'erreur moyenne en LUV pour les cinq méthodes par un graphique. Ces graphiques présentent le logarithme naturel de l'erreur moyenne en LUV par rapport au nombre de correspondances. Chaque type de pointillé représente une transformation de couleurs.

La figure 4.14 montre le graphique pour la *rotation* de la figure 4.2. On voit qu'il y a deux catégories de transformations. Les transformations en RGB réussissent à retrouver l'image objectif même si la transformation quadratique exige plus de correspondances pour parvenir à un résultat comparable à la transformation linéaire en RGB. Par contre, les transformations dans l'espace LUV sont très loin de l'objectif comme on l'a vu précédemment. L'ajout de correspondances n'aidra pas vraiment puisque les transformations en LUV sont assez stables dans les derniers 30 points. La figure 4.20 présente le résultat des cinq transformations pour la rotation avec translation à l'aide des 50 correspondances qui ont servi pour le graphique. Le tableau 4.13 indique la moyenne, la variance et le maximum de l'erreur moyenne en LUV de 50 ensembles différents de 50 associations pour les cinq transformations.

Transformations	Moyenne	Variance	Maximum
Linéaire en RGB	0.197	0.000175	0.224
Quadratique en RGB	0.259	0.00559	0.528
Linéaire en (Luv)	19.2	0.973	22.0
Linéaire en (L) et (uv)	39.6	6.74	47.5
Linéaire en (L) et cubique en (uv)	39.5	17.95	50.3

TAB. 4.13: L'influence de la sélection des points pour la fig. 4.2 : *rotation* pour toutes les transformations. Statistiques sur l'erreur moyenne en LUV pour 50 essais avec 50 correspondances prises au hasard.

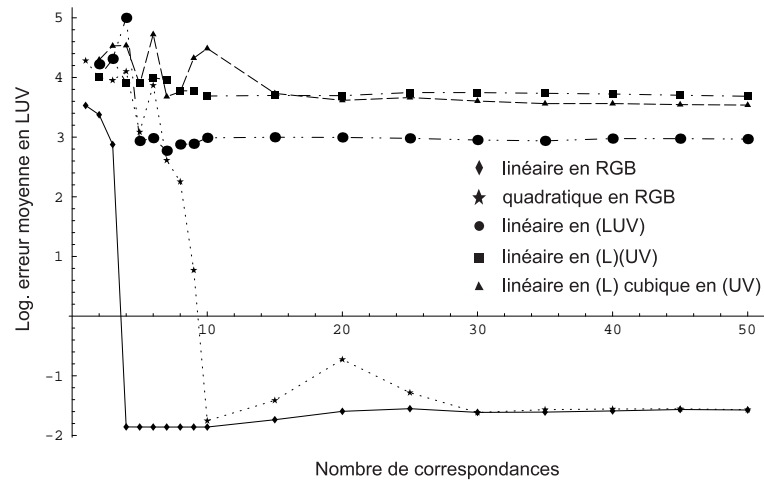


FIG. 4.14: Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la figure 4.2 : *rotation*.

Le graphique de la figure 4.15 donne le logarithme de l'erreur en LUV par rapport au nombre de correspondances pour l'exemple du *cisaillement* de la figure 4.3. Le tableau 4.14 donne les mêmes informations sur l'erreur moyenne du *cisaillement* que le tableau précédent. Cette fois, puisqu'il n'y a pas de rotation dans la transformation, les transformations en LUV réussissent mieux que pour l'exemple de la *rotation*. Cependant, celles qui traitent séparément la luminance ont une plus grande erreur moyenne que les autres (environ trois en LUV comparativement à moins de un pour les autres). Les trois autres transformations obtiennent des résultats équivalents entre elles avec 50 correspondances, mais ce n'est pas nécessaire d'avoir le même nombre de points pour obtenir ces résultats. Avec sept correspondances, les résultats de la transformation linéaire en RGB sont déjà en-dessous de l'unité pour l'erreur moyenne en LUV, tandis que la transformation quadratique en RGB et linéaire en (*Luv*) en prennent 15. La figure 4.21 présente les résultats des cinq transformations sur l'exemple du *cisaillement* avec les 50 correspondances servant au graphique.

Le graphique de la figure 4.16 présente les variations du logarithme de l'erreur en LUV en fonction du nombre de correspondances pour l'exemple de la *compression* du haut de la figure 4.13. Le tableau 4.15 donne la moyenne, la variance et le maximum de l'erreur moyenne en LUV pour cet exemple. La figure 4.22 montre les résultats des transformations de couleurs avec 50 associations. Pour cet exemple, les deux transformations en RGB s'équivalent. La transformation linéaire en RGB donne en moyenne

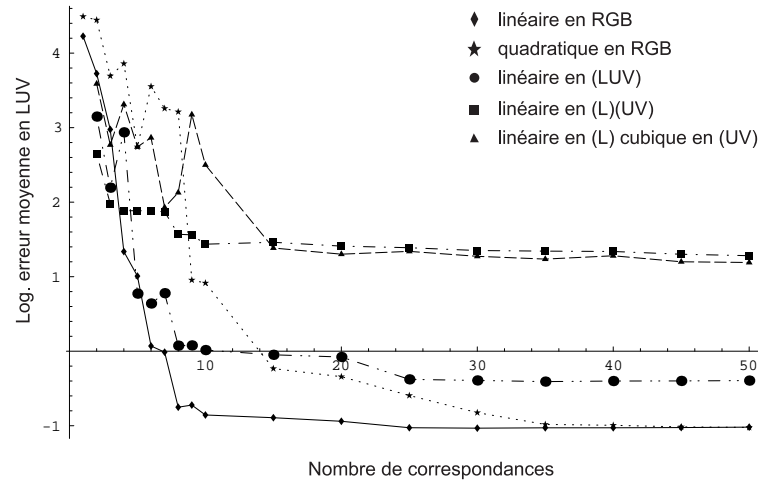


FIG. 4.15: Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la figure 4.3 : *cisaillement*.

Transformations	Moyenne	Variance	Maximum
Linéaire en RGB	0.368	0.000141	0.414
Quadratique en RGB	0.420	0.00297	0.579
Linéaire en (<i>Luv</i>)	0.644	0.00698	0.924
Linéaire en (<i>L</i>) et (<i>uv</i>)	3.51	0.0401	4.26
Linéaire en (<i>L</i>) et cubique en (<i>uv</i>)	3.64	0.486	5.94

TAB. 4.14: L'influence de la sélection des points pour la fig. 4.3 : *cisaillement* pour toutes les transformations. Statistiques sur l'erreur moyenne en LUV pour 50 essais avec 50 correspondances prises au hasard.

une erreur légèrement plus grande que celle de la quadratique (2.29 contre 2.22), mais celle-là a un maximum plus petit (2.55 contre 2.96) avec moins de variations. Puisque la transformation à trouver est presque linéaire, la transformation quadratique peut obtenir des “meilleurs” résultats étant elle-même non-linéaire. Mais le degré de liberté supplémentaire peut être décevant et donner une erreur moyenne supérieure à celle obtenue par la transformation linéaire en RGB. La transformation linéaire en (*Luv*) est assez bonne (2.59), mais l'échange des couleurs ne lui permet pas d'avoir des résultats semblables aux transformations en RGB. Les deux dernières transformations n'ont pas assez de liberté avec la séparation de la luminance pour se rapprocher des résultats des autres transformations.

Le graphique de la figure 4.17, le tableau 4.16 et la figure 4.23 donnent les résultats sous diverses formes pour l'exemple de la transformation *gamma*. Comme cet exemple est non-linéaire, il est normal que les transformations non-linéaires réussissent mieux à

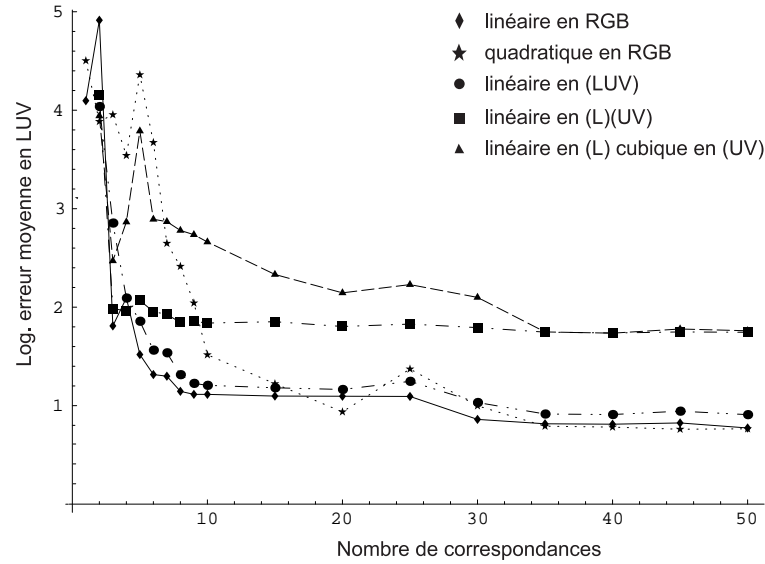


FIG. 4.16: Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur le haut de la figure 4.13 : *compression*.

Transformations	Moyenne	Variance	Maximum
Linéaire en RGB	2.29	0.00850	2.55
Quadratique en RGB	2.22	0.0368	2.96
Linéaire en (<i>Luv</i>)	2.59	0.0350	3.22
Linéaire en (<i>L</i>) et (<i>uv</i>)	5.64	0.0196	6.12
Linéaire en (<i>L</i>) et cubique en (<i>uv</i>)	5.81	0.178	7.86

TAB. 4.15: L'influence de la sélection des points pour la fig. 4.13 en haut : *compression* pour toutes les transformations. Statistiques sur l'erreur moyenne en LUV pour 50 essais avec 50 correspondances prises au hasard.

cerner cette transformation. La transformation quadratique en RGB en est un exemple. Grâce à sa partie cubique, la transformation linéaire en (*L*) et cubique en (*uv*) est légèrement “meilleure” que les trois dernières transformations.

Le graphique de la figure 4.18, le tableau 4.17 et la figure 4.24 présentent les résultats comparatifs pour l'exemple de l'*inconstance de la caméra*. Les correspondances de cet exemple sont prises au hasard mais pas dans la zone du vase et de son ombre. Les calculs d'erreur sont faits sur la région illustrée par la figure 4.8 pour que la différence de scène ne vienne pas interférer dans les résultats. La “meilleure” transformation pour cet exemple est la linéaire en (*Luv*). L'espace LUV est mieux adapté pour l'*inconstance de la caméra*, mais la transformation linéaire en RGB n'est pas loin derrière (2.77 contre 2.69). Dans cet exemple, les transformations de degré plus élevé comme la quadra-

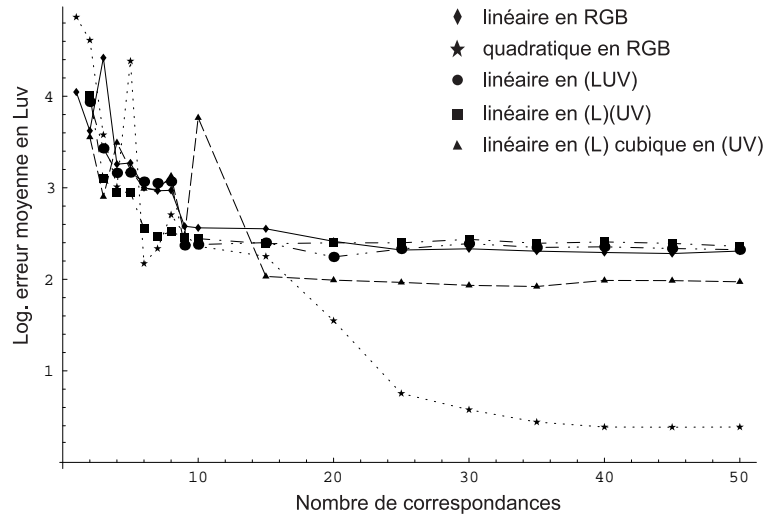


FIG. 4.17: Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la figure 4.6 : *gamma*.

Transformations	Moyenne	Variance	Maximum
Linéaire en RGB	10.50	0.270	11.92
Quadratique en RGB	1.55	0.0146	1.87
Linéaire en (<i>Luv</i>)	10.98	3.53	17.68
Linéaire en (<i>L</i>) et (<i>uv</i>)	12.63	2.98	17.68
Linéaire en (<i>L</i>) et cubique en (<i>uv</i>)	7.87	1.65	11.70

TAB. 4.16: L'influence de la sélection des points pour la fig. 4.6 : *gamma* pour toutes les transformations. Statistiques sur l'erreur moyenne en LUV pour 50 essais avec 50 correspondances prises au hasard.

tique en RGB et la linéaire en (*L*) et cubique en (*uv*) sont moins bien adaptées parce qu'elles demandent beaucoup de points avant que les résultats se stabilisent. En effet, les couleurs sont tellement linéairement dépendantes les unes des autres qu'il faut plusieurs correspondances pour obtenir assez de couleurs linéairement indépendantes pour éliminer les artefacts. On peut le remarquer à l'aide du tableau 4.18 qui présente les mêmes statistiques que le tableau 4.17 mais avec des ensembles de 20 correspondances. Ces deux transformations ont une moyenne, une variance et un maximum beaucoup plus élevés que les autres transformations.

Le graphique 4.19, le tableau 4.19 et la figure 4.25 montrent les résultats des différentes transformations pour l'exemple du *changement d'éclairage* du bas de la figure 4.13. Pour cet exemple, la "meilleure" transformation semble être la transformation quadratique en RGB, mais il faut faire attention parce qu'il y a un risque d'arte-

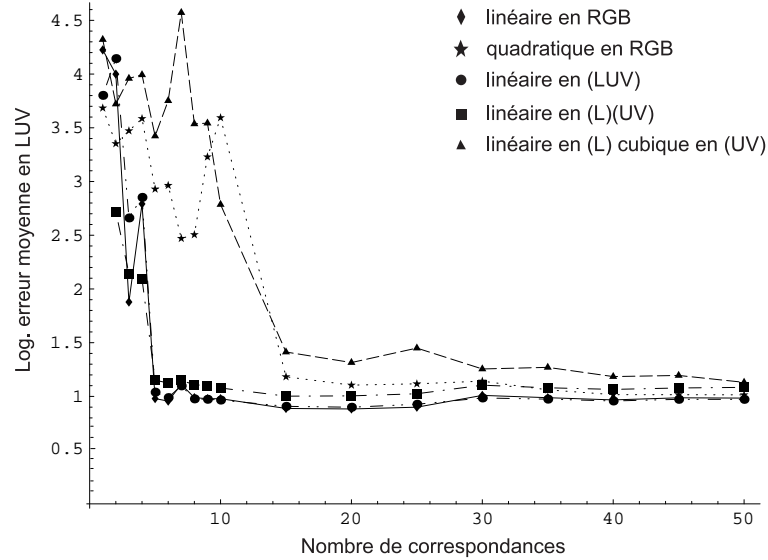


FIG. 4.18: Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur la figure 4.7 : *inconstance de la caméra*.

Transformations	Moyenne	Variance	Maximum
Linéaire en RGB	2.77	0.0430	3.44
Quadratique en RGB	3.10	0.113	4.45
Linéaire en (Luv)	2.69	0.0222	3.19
Linéaire en (L) et (uv)	3.01	0.0346	3.54
Linéaire en (L) et cubique en (uv)	3.22	0.0967	4.36

TAB. 4.17: L'influence de la sélection des points pour la fig. 4.7 : *inconstance de la caméra* pour toutes les transformations. Statistiques sur l'erreur moyenne en LUV sur une partie de l'image pour 50 essais avec 50 correspondances prises au hasard, mais hors de la zone du vase et de son ombre.

fact. Comme dans l'exemple précédent, la gamme de couleurs présentes dans les deux images de départ est très alignée. Certains ensembles de correspondances donneront un "meilleur" résultat avec la transformation linéaire en (Luv) qu'avec la quadratique en RGB puisque le maximum de l'erreur moyenne est de 12.1 avec la quadratique et de 10.1 avec la linéaire en (Luv). Les autres transformations réussissent moins bien que les deux précédentes.

Dans ce chapitre, nous avons comparé cinq transformations différentes : transformation linéaire en RGB, quadratique en RGB, linéaire en (Luv), linéaire en (L)(uv) et, enfin, linéaire en (L) et cubique en (uv). Les transformations décrites ne sont que des

Transformations	Moyenne	Variance	Maximum
Linéaire en RGB	3.10	0.280	5.58
Quadratique en RGB	5.20	5.30	15.14
Linéaire en (Luv)	2.99	0.127	4.32
Linéaire en (L) et (uv)	3.15	0.0758	3.94
Linéaire en (L) et cubique en (uv)	6.44	16.74	27.39

TAB. 4.18: L'influence de la sélection des points pour la fig. 4.7 : *inconstance de la caméra* pour toutes les transformations. Statistiques sur l'erreur moyenne en LUV sur une partie de l'image pour 50 essais avec 20 correspondances prises au hasard, mais hors de la zone du vase et de son ombre.

Transformations	Moyenne	Variance	Maximum
Linéaire en RGB	8.80	1.54	12.3
Quadratique en RGB	5.69	2.03	12.1
Linéaire en (Luv)	7.90	0.686	10.1
Linéaire en (L) et (uv)	10.38	1.32	13.9
Linéaire en (L) et cubique en (uv)	10.65	8.07	19.1

TAB. 4.19: L'influence de la sélection des points pour la fig. 4.13 en bas : *changement d'éclairage* pour toutes les transformations. Statistiques sur l'erreur moyenne en LUV pour 50 essais avec 50 correspondances prises au hasard.

exemples de toutes les transformations de couleurs possibles. Cette exploration n'est certes pas exhaustive. Cependant, nous avons observé qu'une augmentation de degré des équations de transformations semble créer plus d'artefacts et exiger beaucoup plus de points pour atteindre la stabilité des résultats. On a aussi vu que la séparation de la luminance des composantes chromatiques n'apporte aucune amélioration. Pour des images de départ séparées par une transformation de couleurs linéaire en RGB ou presque, une transformation linéaire est amplement suffisante et donne de très bons résultats avec peu de points. Lorsqu'une transformation hautement non-linéaire sépare les images de départ, la transformation quadratique a assez de liberté pour cerner plus convenablement certaines transformations non-linéaires. Et enfin, pour les changements d'éclairage et les différences de luminance, l'utilisation de la transformation linéaire en (Luv) semble plus appropriée. Malgré ces résultats, l'utilisation de d'autres transformations peut s'avérer plus efficace que celles discutées ici. Présentement, chaque association a la même importance que les autres. On pourrait faire correspondre un poids à chaque association de façon à favoriser certaines associations. L'utilisateur pourrait ainsi contraindre certaines couleurs à devenir très proches de la couleur associée au détriment

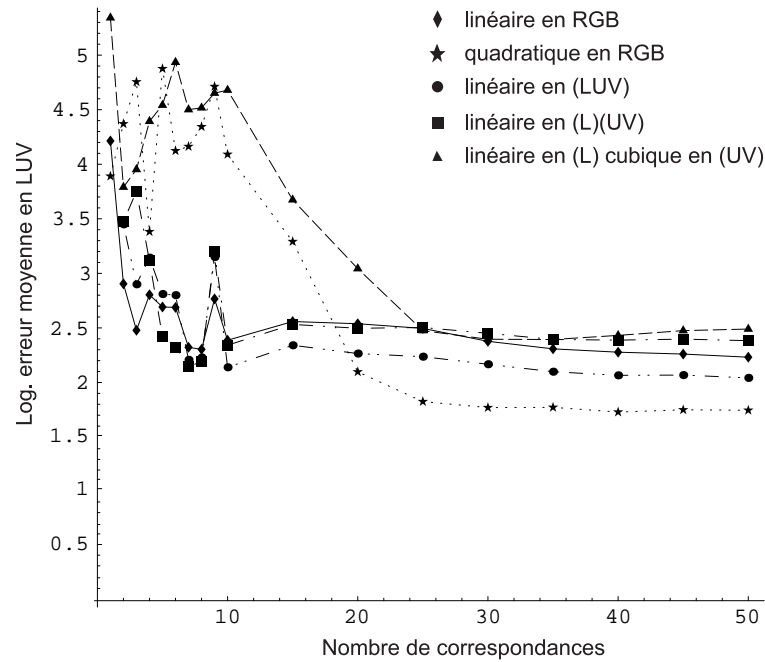


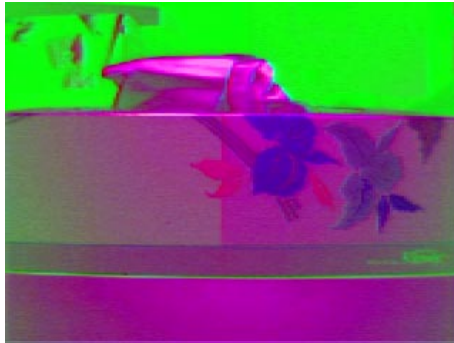
FIG. 4.19: Graphique du log. de l'erreur moyenne en LUV pour les cinq transformations sur le bas de la figure 4.13 : *changement d'éclairage*.

de couleurs moins importantes.

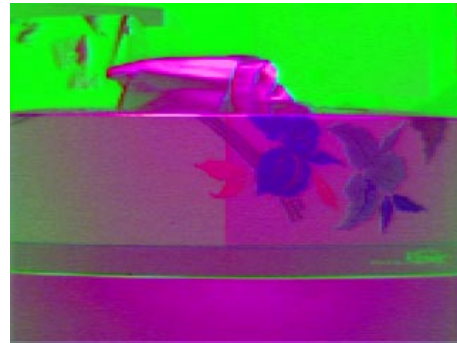
Mais toutes ces transformations ont besoin d'une mise en correspondance des couleurs des deux images de départ. Les exemples utilisés dans ce chapitre sont formés de scènes très semblables, ce qui permet de choisir les points au hasard dans la première image et de lui faire correspondre le pixel ayant la même position dans la seconde image. Le choix des ensembles de points se fait assez rapidement. Cependant, quand l'utilisateur doit choisir lui-même 50 associations, il risque de trouver cela monotone. De plus, étant humain, l'utilisateur peut se tromper en associant des couleurs ensemble. Pour ces raisons, le chapitre suivant aborde l'exploration de mise en correspondance de couleurs de façon plus automatique.



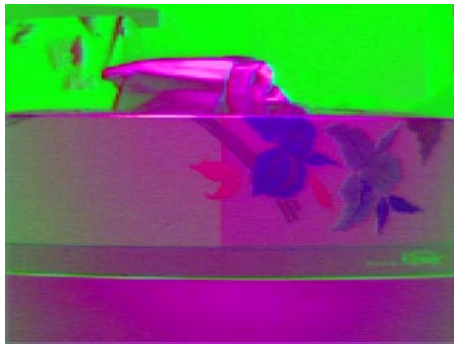
a) Image de départ



b) Image objectif



c) Linéaire en RGB



d) Quadratique en RGB

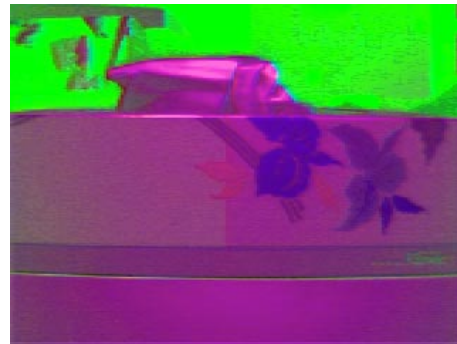
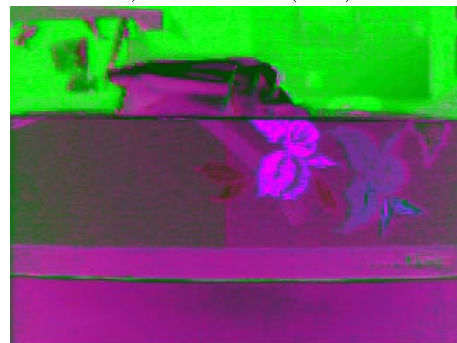
e) Linéaire en (Luv) f) Linéaire en (L) et en (uv) g) Linéaire en (L) et cubique en (uv)

FIG. 4.20: Comparaison des résultats des différentes transformations pour retrouver la *rotation*. L'image a) est celle de départ. L'image b) est l'objectif recherché. Les images c) à g) sont les résultats des différentes transformations avec la même série de 50 correspondances.



a) Image de départ



b) Image objectif



c) Linéaire en RGB



d) Quadratique en RGB

e) Linéaire en (Luv) f) Linéaire en (L) et en (uv) g) Linéaire en (L) et cubique en (uv)

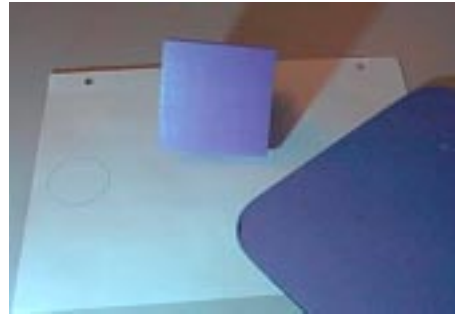
FIG. 4.21: Comparaison des résultats des différentes transformations pour retrouver le *cisaillement*. L'image a) est celle de départ. L'image b) est l'objectif recherché. Les images c) à g) sont les résultats des différentes transformations avec la même série de 50 correspondances.



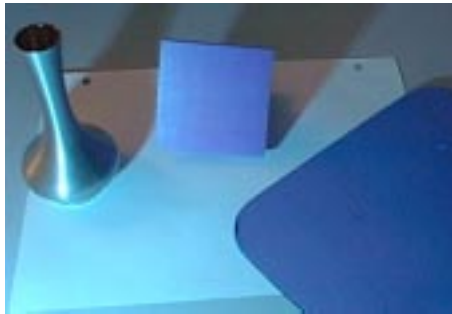
FIG. 4.22: Comparaison des résultats des différentes transformations pour retrouver la *compression*. L'image a) est celle de départ. L'image b) est l'objectif recherché. Les images c) à g) sont les résultats des différentes transformations avec la même série de 50 correspondances.



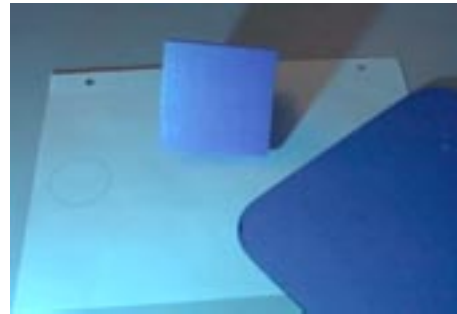
FIG. 4.23: Comparaison des résultats des différentes transformations pour retrouver la transformation *gamma* 2.5. L'image a) est celle de départ. L'image b) est l'objectif recherché. Les images c) à g) sont les résultats des différentes transformations avec la même série de 50 correspondances.



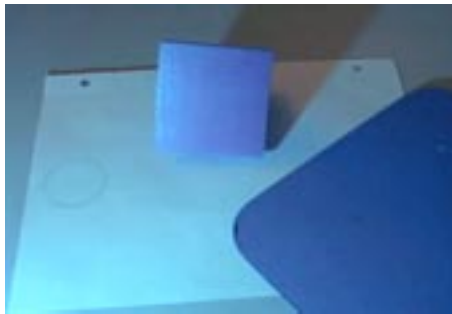
a) Image de départ



b) Image objectif



c) Linéaire en RGB



d) Quadratique en RGB

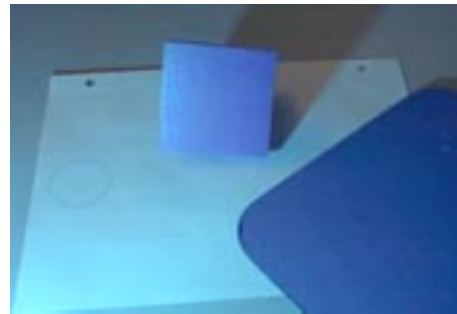
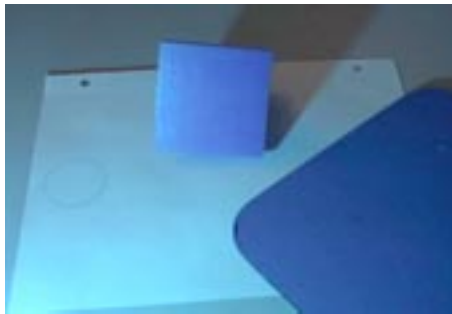
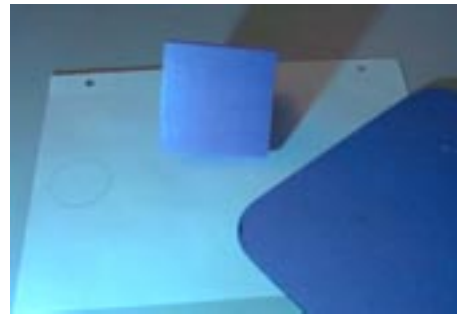
e) Linéaire en (Luv) f) Linéaire en (L) et en (uv) g) Linéaire en (L) et cubique en (uv)

FIG. 4.24: Comparaison des résultats des différentes transformations pour corriger l'*inconstance de la caméra*. L'image a) est celle de départ. L'image b) est l'objectif recherché. Les images c) à g) sont les résultats des différentes transformations avec la même série de 50 correspondances.

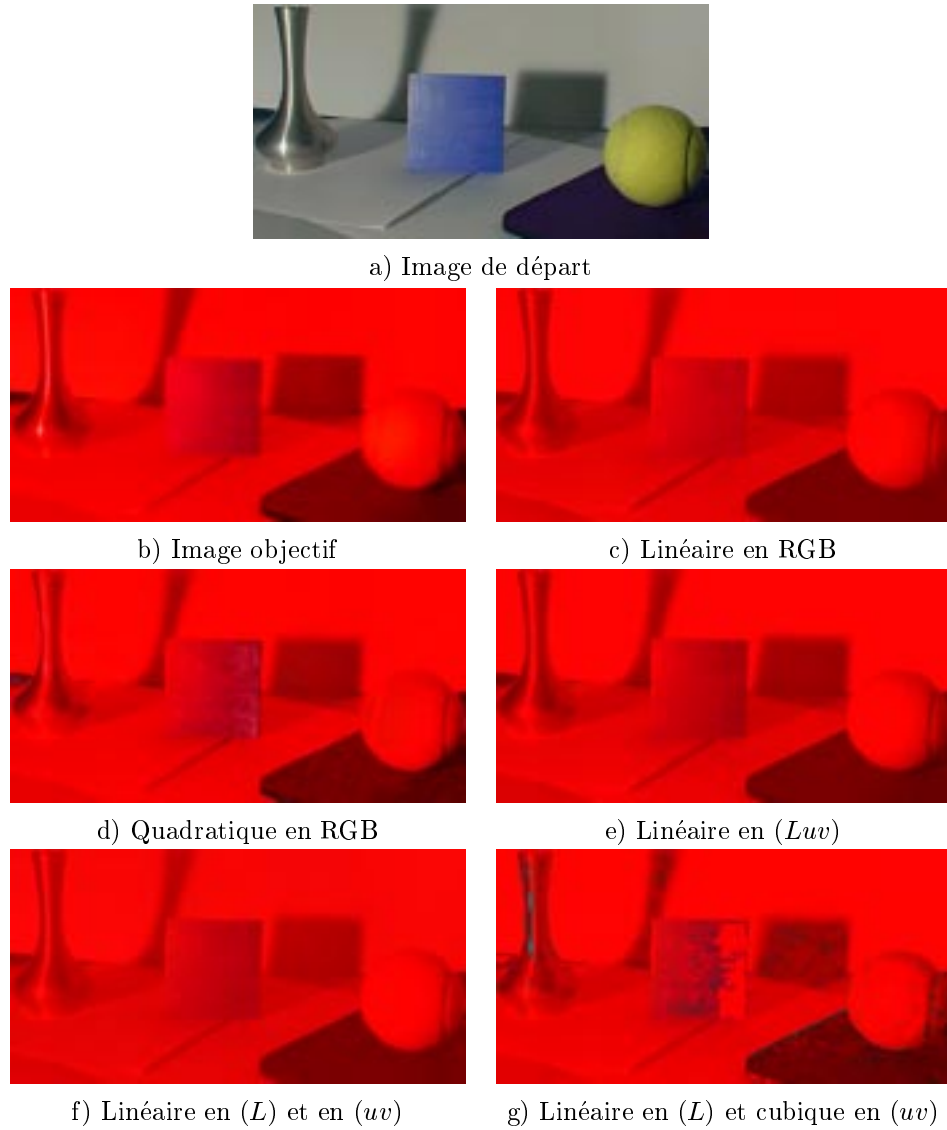


FIG. 4.25: Comparaison des résultats des différentes transformations pour retrouver le *changement d'éclairage*. L'image a) est celle de départ. L'image b) est l'objectif recherché. Les images c) à g) sont les résultats des différentes transformations avec la même série de 50 correspondances.

Chapitre 5

Association automatique des couleurs

Dans le chapitre précédent, nous avons vu une méthode pour mettre en correspondance des couleurs. Cette méthode est totalement contrôlée par l'utilisateur puisque c'est lui qui associe le nombre de couleurs qu'il veut d'une image à l'autre. Cette méthode a des désavantages. Elle peut être longue et fastidieuse si l'utilisateur désire beaucoup de correspondances. Dans certaines circonstances, le choix des couleurs peut être plus difficile à effectuer par l'utilisateur. Par exemple, dans une région où les couleurs changent radicalement d'un pixel à l'autre (hautes fréquences), l'utilisateur peut éprouver des difficultés à sélectionner un pixel précis et non ses voisins. La figure 5.1 montre deux régions de ce type : une texture de bois et du feuillage. Un deuxième type de régions difficiles à traiter est une région ayant peu de différences entre les couleurs. Un exemple de ce type est une région contenant de la peau. Il est très difficile de choisir une dizaine de points dans l'image de droite de la figure 5.1 et de les associer à des couleurs d'une autre image de peau sans effectuer de mauvaises correspondances.



a) texture de bois



b) feuillage



c) peau

FIG. 5.1: Images où la mise en correspondance est difficile pour l'utilisateur.

Pour contrer ces désavantages, nous avons étudié des méthodes de mise en correspondance plus automatiques où l'utilisateur désigne des régions au lieu de points. Dans ce chapitre, nous verrons deux méthodes d'association de régions et ensuite ce que nous pouvons faire lorsque plusieurs associations de régions sont impliquées dans la transformation.

5.1 Méthodes d'association de régions

Au lieu de choisir des pixels, l'utilisateur peut choisir des régions. L'utilisateur désigne une région dans la première image et une seconde dans la deuxième. C'est à l'algorithme de choisir parmi toutes les couleurs des régions celles qui doivent être mises en correspondance. Après la mise en correspondance des couleurs, on applique une des transformations présentées au chapitre précédent. À la figure 5.2, on a une correspondance de régions. Cette correspondance de régions signifie qu'on veut transformer les couleurs de la région de l'image de départ pour qu'elles deviennent semblables aux couleurs de la région désignée dans l'image objectif. Pour des fins de simplicité d'interface, notre sélection de région ne considère présentement que des régions rectangulaires.



FIG. 5.2: Mise en correspondance de régions par l'utilisateur.

Il reste à définir comment l'algorithme "choisit" les couleurs pour la mise en correspondance. Nous avons développé deux stratégies : une basée sur les statistiques et une autre sur les vecteurs propres des couleurs des régions.

5.1.1 Méthode statistique

La méthode statistique est basée sur l'ordonnance des couleurs. Dans un premier temps, on ordonne les couleurs de chaque région séparément. Les espaces couleurs considérés ici ont tous trois dimensions. Le critère que nous avons choisi est la luminance. On classe toutes les couleurs d'une région par ordre croissant de luminance. Une fois que les couleurs des deux régions sont ordonnées, on prend un certain nombre de centiles répartis sur toute la distribution. On associe ensemble la couleur correspondant au centile de chaque distribution des régions. Pour l'implantation, nous avons pris 15 correspondances. Nous avons choisi les couleurs ayant la luminance la plus faible et la plus forte ainsi que 13 centiles équitablement répartis sur toute la distribution. Un exemple de correspondances est illustré par le schéma suivant :

$$\begin{array}{ccccccccccc} \text{distribution1 : } & L_{1_1} & \leq & L_{1_2} & \leq & \dots & \leq & L_{1_{39}} & \leq & \dots & \leq & L_{1_{77}} & \leq & L_{1_{78}} \\ & \downarrow & & & & & & \downarrow & & & & & \downarrow & \\ \text{distribution2 : } & L_{2_1} & \leq & L_{2_2} & \leq & \dots & \leq & L_{2_{46}} & \leq & \dots & \leq & L_{2_{91}} & \leq & L_{2_{92}} \end{array} .$$

Dans cet exemple, supposons que les régions ont 78 et 92 pixels. Les *distribution1* et *distribution2* sont les ensembles ordonnés de la luminance des pixels des régions de l'image de départ et de celle objectif. Les flèches symbolisent la correspondance entre les couleurs qui ont cette luminance. Nous savons que cette façon de classer les couleurs n'est pas parfaite puisqu'en prenant seulement la luminance comme critère, il y a une perte d'information (la chromaticité). Mais tous les critères permettant d'ordonner selon une dimension des objets en trois dimensions peuvent produire une perte d'information.

Résultats

Pour un premier exemple, on a repris les images de l'exemple de la *compression* en sélectionnant une région contenant du ciel et de la mer. La région correspondante dans la seconde image est positionnée à la même place puisque les scènes sont identiques. À la figure 5.3, les régions sélectionnées dans les images de départ et objectif sont les plus grands rectangles. À la figure 5.3, on voit également le résultat de la correspondance par la méthode statistique avec une transformation linéaire en RGB. Puisque les couleurs

prises en correspondance sont prises seulement dans les régions associées, on ne peut pas s'attendre à obtenir un bon résultat sur toute l'image. Pour cette raison, les calculs de l'erreur sont faits seulement sur la même portion de l'image que les régions sélectionnées. L'erreur moyenne sur cette région est de 3.78 en LUV et la variance de 5.28 en LUV.



FIG. 5.3: Correspondance de régions avec la méthode statistique et une transformation linéaire en RGB. En haut : l'image de départ. En bas à gauche : l'image objectif. En bas à droite : le résultat obtenu. L'erreur est calculée seulement sur la région désignée sur l'image de départ. Les grands rectangles montrent les régions sélectionnées. Le petit rectangle va servir de région incluse dans les 50 régions prises au hasard pour démontrer l'influence du choix des régions. Les images ont 420 par 280 pixels.

Pour montrer l'influence du choix des régions, on a pris au hasard 50 régions comprises dans le grand rectangle de l'image objectif de la figure 5.3 mais comprenant le petit rectangle. La petite région sera utilisée pour le calcul de l'erreur. La moyenne est de 3.14, la variance est de 1.03 et le maximum de l'erreur moyenne en LUV sur la région est de 6.23.

Cette méthode a par contre quelques défauts. Les couleurs des régions qu'on associe doivent apparaître dans les mêmes proportions dans les deux régions. Dans l'exemple précédent, les régions contiennent un regroupement de couleurs pour le ciel et un autre pour la mer. Si les régions associées n'ont pas les mêmes proportions de ciel et de mer, une couleur appartenant au ciel dans une image peut se retrouver associée à une couleur

de la mer dans l'autre image. La figure 5.4 montre des régions choisies ne contenant pas les mêmes proportions et le résultat obtenu. L'erreur est calculée sur la même région que celle sélectionnée dans l'image de départ. L'erreur moyenne est de 8.18 en LUV avec une variance de 34.0 comparativement à 3.78 en moyenne et de 5.28 pour la variance pour la figure 5.3.



FIG. 5.4: Correspondance de régions ayant des proportions différentes avec la méthode statistique et une transformation linéaire en RGB. En haut : l'image de départ. En bas à gauche : l'image objectif. En bas à droite : le résultat obtenu. L'erreur est calculée seulement sur la région désignée sur l'image de départ.

La figure 5.5 montre ce qui se passe lorsque les couleurs ne sont pas dans les mêmes proportions d'une région à l'autre. Cette figure représente le cube RGB. Les couleurs de la région de l'image de départ sont représentées en gris pâle. Celles de la région de l'image objectif sont en gris foncé. Les lignes blanches relient les couleurs que la méthode statistique a associées ensemble. Comme on peut voir, les deux régions possèdent chacun deux groupes de couleurs distinctes. Si les couleurs étaient dans les mêmes proportions dans les deux régions, les lignes blanches relieraient les couleurs d'un groupe d'une région aux couleurs d'un des deux groupes de l'autre région. Ici, les couleurs d'un même groupe sont associées à des couleurs n'appartenant pas au même groupe. Dans notre exemple, cela signifie que certaines couleurs du ciel de l'image de départ sont associées aux couleurs du ciel de l'image objectif, mais que d'autres sont associées aux

couleurs de la mer.

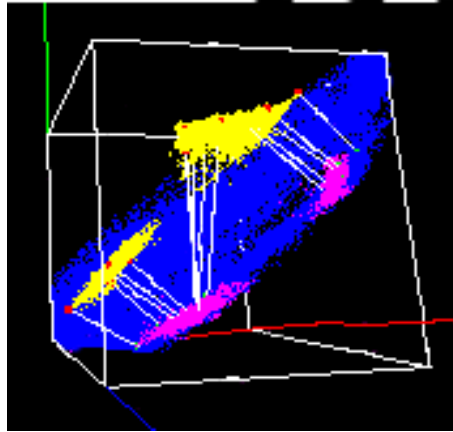


FIG. 5.5: Cube RGB montrant les associations lorsque les régions n'ont pas les mêmes proportions. En gris pâle : les couleurs de la région de l'image de départ. En gris foncé : les couleurs de la région de l'image objectif. Les traits blancs relient les couleurs associées.

En associant les couleurs des différents centiles des distributions de couleurs, on associe toujours ensemble les couleurs ayant la plus grande luminance de chaque distribution ainsi que celles qui ont la plus faible luminance. Dès que l'ordre des couleurs par la luminance n'est pas identique dans les deux images de départ, l'association des couleurs par la méthode statistique ne fonctionnera pas. Comme on l'a vu au chapitre précédent, les transformations de rotation du cube RGB sont pratiquement exclues. En fait, la rotation du cube peut ne pas modifier l'ordre des couleurs, cela dépend de la disposition de la distribution. À la figure 5.6, on a repris l'exemple de la *rotation*. L'image de gauche est l'image objectif. Un rectangle montre la région choisie dans les deux images de départ. L'image de droite est le résultat obtenu. L'erreur calculée dans le même rectangle est de 75.9 en LUV et de 4400 pour la variance.

D'un autre côté, s'il n'y a pas assez de différences entre les couleurs de la région choisie, des artefacts comme des taches peuvent se former. Si les couleurs sont presque sur une droite, le risque est encore plus grand. Comme nous le verrons plus en détail au chapitre 6, sous une seule source de lumière, les couleurs d'un objet diélectrique ayant une couleur uniforme forment deux vecteurs dans le cube RGB [Kli93] [NS92]. Les couleurs de la partie de l'objet qui réfléchit de façon plus diffuse que spéculaire forme un vecteur orienté selon la droite passant par le noir et la couleur de l'objet. Les couleurs de la partie plus spéculaire de l'objet forment un second vecteur qui part

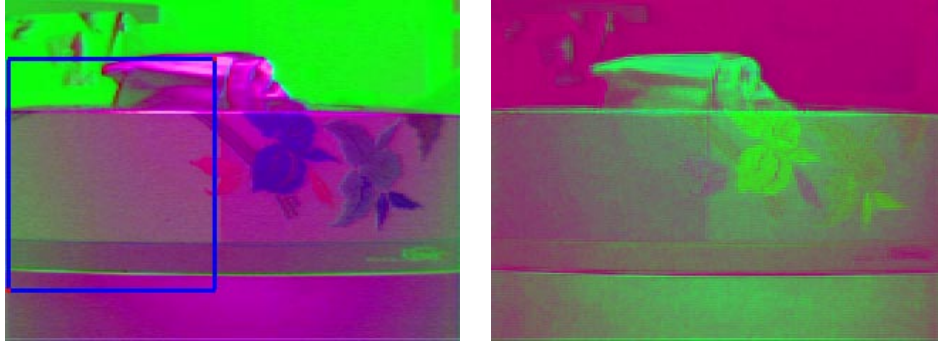


FIG. 5.6: Correspondance de régions par la méthode statistique avec une transformation linéaire en RGB sur l'exemple de la *rotation*. À gauche : l'image objectif avec le rectangle désignant la région choisie. À droite : le résultat obtenu.

d'un point du premier vecteur et qui se dirige vers la couleur de la lumière. Donc, si la région qu'on choisit contient seulement un objet de couleur uniforme, on risque de créer des artefacts. La figure 5.7 montre la même scène photographiée sous deux éclairages différents : une illuminée par une ampoule de 100 watts blanche et l'autre par une ampoule rouge de 60 watts. Les rectangles désignent les régions sélectionnées. L'erreur est calculée sur la même région. L'erreur moyenne est de 2.39 en LUV et la variance de 1.36. Bien que l'erreur moyenne et la variance soient assez petites, on voit nettement des taches sur la région transformée.

Comme on vient de le voir, il y a beaucoup de cas où la méthode statistique ne fonctionne pas très bien pour faire les correspondances entre les couleurs. Pour cette raison, nous avons cherché une autre méthode.

5.1.2 Méthode vectorielle

La méthode vectorielle utilise la technique d'Hotelling¹ [GW92]. Cette technique est particulièrement employée en reconnaissance de formes. Avant de reconnaître la forme d'un objet en deux dimensions, il faut aligner l'objet testé sur le modèle. La méthode d'Hotelling permet de faire cet alignement. Le principe est simple : on trouve les deux axes principaux orthogonaux de l'objet et du modèle et on calcule la transformation permettant aux axes de l'objet de se superposer aux axes du modèle. Pour la transformation de couleurs, les objets à aligner sont les nuages de points de couleurs des régions

¹Cette technique est aussi connue sous les noms de transformation par les vecteurs propres, par composantes principales ou la transformation discrète de Karhunen-Loève.

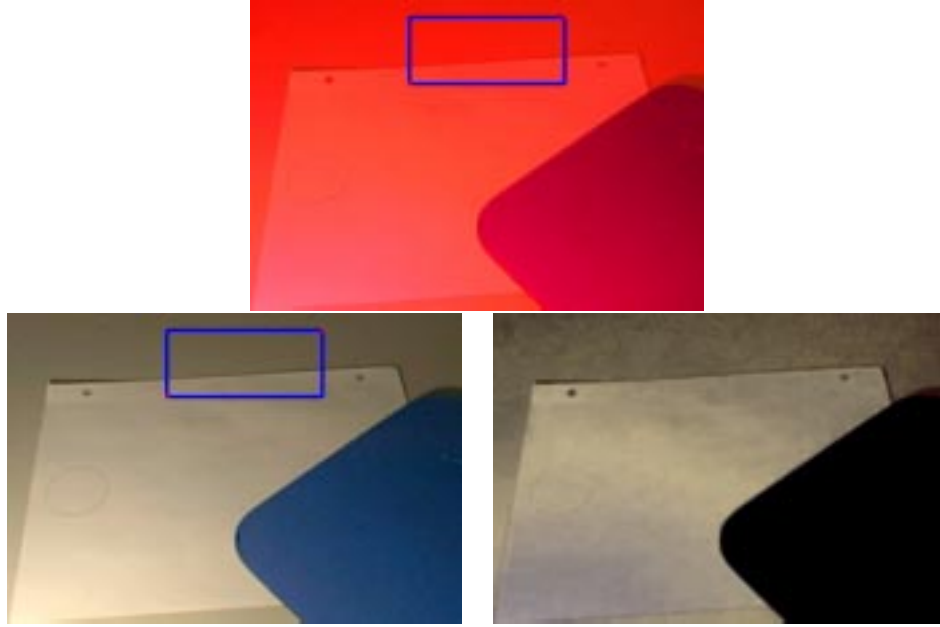


FIG. 5.7: Correspondance de régions par la méthode statistique avec une transformation linéaire en RGB pour un exemple où il y a peu de changements de couleurs. En haut : l'image de départ avec la région sélectionnée. À gauche : l'image objectif avec la région choisie. À droite : le résultat obtenu. Les images ont 380 par 260 pixels.

choisies dans l'espace RGB. Ces nuages de points sont en trois dimensions contrairement aux objets à reconnaître dans une image en deux dimensions. Chaque pixel de la région choisie sera utilisé sous la forme d'un vecteur couleur de trois dimensions : r, g et b . On calcule le vecteur couleur moyen et la matrice de covariance des vecteurs couleurs. Les équations pour la moyenne et la covariance sont les suivantes :

$$Moy = \begin{pmatrix} m_r \\ m_g \\ m_b \end{pmatrix} = \frac{1}{N} \sum_{i=1}^N \begin{pmatrix} r_i \\ g_i \\ b_i \end{pmatrix} \quad (5.1)$$

$$Cov = \frac{1}{N} \sum_{i=1}^N \left[\begin{pmatrix} r_i \\ g_i \\ b_i \end{pmatrix} \begin{pmatrix} r_i & g_i & b_i \end{pmatrix} \right] - \begin{pmatrix} m_r \\ m_g \\ m_b \end{pmatrix} \begin{pmatrix} m_r & m_g & m_b \end{pmatrix}. \quad (5.2)$$

Dans ces équations, r_i, g_i et b_i sont les trois composantes du i^e vecteur couleur et N le nombre de vecteurs couleurs.

Les axes principaux du nuage de couleur sont les vecteurs propres de la matrice de covariance. Comme la matrice de covariance est réelle et symétrique, les vecteurs propres de cette matrice sont toujours orthogonaux. On trouve les vecteurs propres de cette matrice en utilisant la méthode des puissances [CC88] pour le premier vecteur

propre, la méthode de déflation [CC88] pour le deuxième et le produit vectoriel des deux premiers vecteurs pour le troisième. On calcule les vecteurs propres et le vecteur moyen pour les deux régions associées. Ensuite, on associe les vecteurs couleurs moyens entre eux. Pour les vecteurs propres, on associe la couleur du point égale au vecteur plus le vecteur moyen. Le vecteur propre correspondant à la valeur propre la plus grande est associé à celui ayant la plus grande valeur propre dans la seconde région et ainsi de suite. Comme un vecteur propre est un représentant de la classe de tous les vecteurs qui sont égaux à un scalaire multiplié par le vecteur propre, il faut décider si le vecteur doit être utilisé dans son sens positif ou négatif. Les vecteurs de la première région sont utilisés dans le sens positif et ceux de la deuxième région sont utilisés dans le sens formant le plus petit angle avec son homologue de l'autre région. Ce choix est tout à fait arbitraire et limite les transformations qu'on peut retrouver. Un exemple de transformation qu'on ne pourrait pas retrouver est un négatif de l'image dont voici le système d'équations :

$$\begin{aligned} r_2 &= 1 - r_1 \\ g_2 &= 1 - g_1 \\ b_2 &= 1 - b_1. \end{aligned} \tag{5.3}$$

Cette méthode permet très bien de retrouver des transformations comme les rotations, les translations ainsi que les changements d'échelle positifs du cube RGB. Cependant, cette méthode ne permet pas de retrouver un cisaillement du cube RGB.

Résultats

Pour permettre de comparer avec la méthode statistique, nous sélectionnons la même région que celle de la figure 5.3 toujours avec l'exemple de la *compression*. À la figure 5.8, on voit à gauche l'image objectif avec le grand rectangle comme région sélectionnée et à droite le résultat de la méthode vectorielle avec une transformation linéaire en RGB. Le petit rectangle sert pour choisir les 50 régions. Les calculs pour l'erreur sont faits sur la même région que celle montrée sur l'image de gauche. L'erreur moyenne sur cette région est de 4.29 en LUV et 8.37 pour la variance comparativement à 3.78 pour la moyenne et 5.28 pour la variance par la méthode statistique.

Pour permettre de bien comparer les résultats des deux méthodes, on a repris les mêmes 50 régions utilisées précédemment. La moyenne est de 2.39, la variance est de



FIG. 5.8: Correspondance de régions avec la méthode vectorielle et une transformation linéaire en RGB pour les mêmes régions que la figure 5.3. À gauche : l'image objectif avec le grand rectangle comme région sélectionnée. À droite : le résultat obtenu. L'erreur est calculée seulement sur le grand rectangle de l'image objectif. Le petit rectangle sert pour choisir les 50 régions.

0.149 et le maximum de l'erreur moyenne en LUV est de 3.27 sur la région du petit rectangle de la figure 5.8 avec la méthode vectorielle.

Les résultats sont “meilleurs” que ceux obtenus par la méthode statistique. Le nombre de groupes de couleurs inclus dans les régions sélectionnées n'a pas d'importance. Avec la méthode statistique, plus on a de groupes différents de couleurs plus on augmente le risque que l'ordre des couleurs ne soit pas le même dans les deux images. La figure 5.9 reprend l'exemple de la *compression* mais cette fois presque toute l'image est sélectionnée pour la région. La région choisie dans l'image de départ est la même que celle identifiée sur l'image objectif qui est à gauche de la figure 5.9. Le résultat de cette association de régions par la méthode vectorielle avec une transformation linéaire en RGB est à droite de la figure. L'erreur est calculée sur toute l'image. L'erreur moyenne est de 2.77 en LUV et la variance de 9.66. À titre de comparaison, l'association par points vue au chapitre précédent avec une transformation linéaire en RGB obtenait en moyenne sur 50 ensembles de 50 correspondances 2.29 comme erreur moyenne.

Avec la méthode vectorielle, le problème des proportions qui doivent être les mêmes d'une région à l'autre est toujours présent. Reprenons les mêmes régions que la figure 5.4. Si, dans une des régions, il y a moins de couleurs de ciel que dans l'autre, alors pour toute proportion gardée, les vecteurs propres vont se rapprocher des couleurs de la mer. La figure 5.10 montre le résultat de la méthode vectorielle avec une transformation linéaire en RGB et des régions n'ayant pas les mêmes proportions de couleurs. L'erreur est calculée selon la région sélectionnée sur l'image de départ qu'on peut voir à la figure 5.4. L'erreur moyenne est de 23.2 en LUV et la variance de 171 comparativement à 8.18



FIG. 5.9: Correspondance de régions avec la méthode vectorielle et une transformation linéaire en RGB avec des régions comprenant beaucoup de couleurs différentes. À gauche : l'image objectif avec la région sélectionnée qui comprend presque toute l'image. À droite : le résultat obtenu. L'erreur est calculée sur toute l'image.

en moyenne et de 34.0 pour la variance avec la méthode statistique.



FIG. 5.10: Correspondance de régions ayant des proportions différentes avec la méthode vectorielle et une transformation linéaire en RGB. À gauche : l'image objectif avec la région choisie. À droite : le résultat obtenu. L'erreur est calculée seulement sur la région désignée sur l'image de départ de la fig. 5.4.

Cependant, le problème causé par les changements dans l'ordre des couleurs pour la méthode statistique ne se présente pas pour la méthode vectorielle sauf si l'ordre est inversé comme l'exemple d'un négatif. Cette méthode retrouve parfaitement les rotations du cube RGB. À la figure 5.11, on a à gauche l'image objectif avec la région choisie. La région correspondante dans l'image de départ est la même. Ces régions sont les mêmes que celles de la figure 5.6. Le résultat de la méthode vectorielle avec une transformation linéaire en RGB est l'image de droite de la figure 5.11. L'erreur est calculée uniquement sur la région désignée sur l'image objectif. L'erreur moyenne est de 0.247 en LUV et la variance est de 0.0399 comparativement à 75.9 de moyenne et 4400 pour la variance avec la méthode statistique.

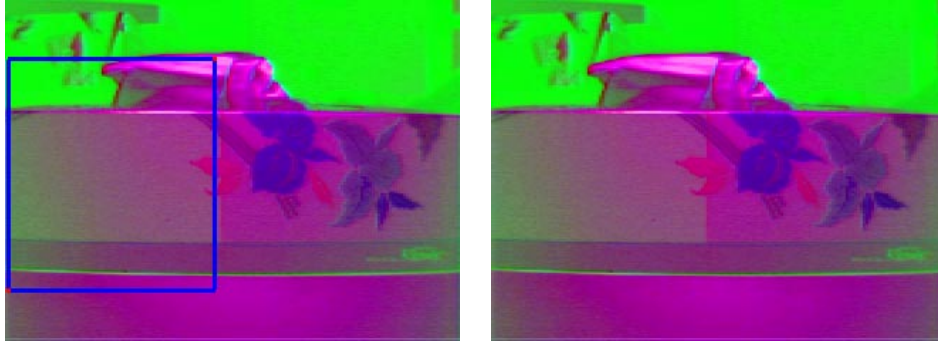


FIG. 5.11: Correspondance de régions par la méthode vectorielle et une transformation linéaire en RGB sur l'exemple de la *rotation*. À gauche : l'image objectif avec un rectangle désignant la région choisie. À droite : le résultat obtenu. L'erreur est calculée sur la même région que celle désignée à l'image objectif.

Comme nous l'avons dit précédemment, la méthode vectorielle ne peut pas retrouver les transformations de cisaillement. À la figure 5.12, nous reprenons l'exemple du *cisaillement*. L'image de gauche est l'image objectif avec la région sélectionnée indiquée par le rectangle. La région choisie dans l'image de départ est la même que celle de l'image objectif. L'image de droite de la figure 5.12 est le résultat de la méthode vectorielle avec une transformation linéaire en RGB et les régions choisies. L'erreur est calculée sur la même région que celle indiquée sur l'image objectif. L'erreur moyenne en LUV est de 3.06 et la variance de 31.4 sur la région. Rappelons que l'utilisateur peut obtenir une erreur moyenne de moins de un en faisant à la main quelques correspondances de couleurs point à point.



FIG. 5.12: Correspondance de régions par la méthode vectorielle et une transformation linéaire en RGB sur l'exemple du *cisaillement*. À gauche : l'image objectif avec un rectangle désignant la région choisie. À droite : le résultat obtenu. L'erreur est calculée sur la même région que celle désignée à l'image objectif.

Le problème des artefacts causés par le peu de variation entre les couleurs a moins d'importance lorsqu'on utilise la méthode vectorielle pour associer des régions. La figure 5.13 reprend les mêmes images que la figure 5.7 et les mêmes régions. À la figure 5.13, l'image objectif est à gauche avec sa région sélectionnée. Le résultat de la méthode vectorielle avec transformation linéaire en RGB est à droite de la figure. L'erreur est calculée sur la même région que la figure 5.7. L'erreur moyenne est de 3.08 en LUV et de 2.35 pour la variance, contrairement à 2.39 pour la moyenne et de 1.36 pour la variance avec la méthode statistique. L'erreur calculée semble indiquer que la méthode statistique est la "meilleure", mais si on regarde bien les figures 5.7 et 5.13, on voit que la méthode statistique donne une surface mouchetée, ce qui n'est pas le cas de la méthode vectorielle. Malgré les erreurs calculées, les résultats obtenus par la méthode vectorielle semblent avoir, visuellement parlant, moins d'artefacts que ceux obtenus par la méthode statistique.



FIG. 5.13: Correspondance de régions par la méthode vectorielle et une transformation linéaire en RGB pour un exemple où il y a peu de changements de couleurs. À gauche : l'image objectif avec un rectangle désignant la région choisie. À droite : le résultat obtenu. L'erreur est calculée sur la même région que celle désignée à l'image objectif.

Nous avons discuté de deux méthodes pour faire des associations de région. Chacune de ces méthodes a ses forces et ses faiblesses. Jusqu'ici nous avons parlé seulement d'associer une région de l'image de départ à une de l'image objectif. Voyons ce qu'on peut faire lorsque plusieurs régions de chaque image sont impliquées.

5.2 Transformation de plusieurs associations de régions

Lorsqu'une région est associée à une autre, les deux méthodes vues précédemment permettent de choisir des correspondances de couleurs. Ensuite, on calcule la transfor-

mation qui se rapproche le plus des contraintes des associations comme on l'a vu au chapitre 4. Puisque les couleurs choisies pour les correspondances sont prises uniquement dans une partie de l'image, l'utilisateur peut vouloir voir l'effet de la transformation seulement sur les couleurs de la région choisie ou sur toute l'image. Le premier cas est appelé l'application locale et le second l'application globale.

5.2.1 Application locale

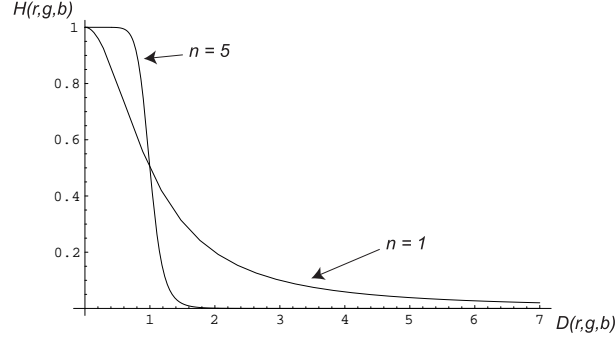
Pour l'application locale, on ne veut pas transformer seulement les pixels positionnés dans la région de l'image de départ. On veut modifier tous les pixels dont la couleur est représentée dans la région choisie. Pour ce faire, on crée une boîte englobante des couleurs de la région dans l'espace RGB. Cette boîte rectangulaire est alignée sur les axes r, g et b . Tous les pixels qui ont leur couleur à l'intérieur de cette boîte sont transformés. Les autres pixels de l'image sont laissés tels quels.

Régions disjointes

Si les boîtes d'une région ou de plusieurs régions dans l'espace RGB sont disjointes, le traitement pour l'application locale est le même. On construit les boîtes pour chaque région et on transforme uniquement les pixels dont la couleur est incluse dans une des boîtes. On applique la transformation calculée pour une région sur toutes les couleurs incluses dans la boîte de cette région.

Régions non-disjointes

Pour des régions non-disjointes, les couleurs incluses dans les boîtes sont affectées par toutes les transformations des régions. Chaque région a sa transformation. On applique chaque transformation sur chaque couleur incluse dans au moins une boîte. De cette façon, on obtient autant de couleurs modifiées que de régions pour chaque couleur à traiter. Chaque couleur ainsi trouvée est ensuite pondérée pour donner la couleur finale. Cette pondération utilise l'équation du filtre passe-bas de Butterworth [GW92] étendue pour les trois dimensions de l'espace RGB :

FIG. 5.14: Variation de la fonction $H(r, g, b)$ lorsque $n = 1$ et 5.

$$H(r, g, b) = \frac{1}{1 + \left(\frac{1}{K} - 1\right) D(r, g, b)^{2n}} \quad (5.4)$$

où

$$D(r, g, b) = \sqrt{\frac{(r - C_r)^2}{D_r^2} + \frac{(g - C_g)^2}{D_g^2} + \frac{(b - C_b)^2}{D_b^2}}. \quad (5.5)$$

Dans cette équation, D_r , D_g et D_b sont les distances séparant le centre de la boîte et ses faces perpendiculaires à l'axe de la composante, K est la valeur de la fonction lorsque le rapport de $D(r, g, b)$ est égal à un, n permet de modifier la pente de la fonction, telle que plus n est grand, plus la pente est abrupte. Le point (C_r, C_g, C_b) est le centre de la boîte et (r, g, b) la couleur à transformer. Dans notre cas, K égale à 10 fois la densité des couleurs (*nombre de couleurs/volume de la boîte*) présentes dans la boîte de couleurs. n peut varier selon les résultats voulus. Un n grand (environ cinq) permet de diminuer l'effet des transformations hors de leur boîte de couleurs. Cependant, l'augmentation de n crée un plateau pour les couleurs à l'intérieur de la boîte comme on peut le voir à la figure 5.14. Ce plateau crée une discontinuité entre les couleurs à l'intérieur de la boîte et celles à l'extérieur. Pour cette raison, on suggère de mettre n à deux.

L'équation de pondération des couleurs est la suivante :

$$\left(\frac{1}{\sum_{i=1}^N H_i(r, g, b)} \right) \sum_{i=1}^N [H_i(r, g, b) \times Ct_i] \quad (5.6)$$

où N est le nombre de régions, H_i est la fonction filtre 5.4 de la i^e région, (r, g, b) est la couleur avant transformation et Ct_i est la couleur (r, g, b) transformée par la transformation de la i^e région. Dans cette pondération, il y a une normalisation pour obtenir une couleur identique aux Ct lorsque la Ct de chaque région est la même.

Avec la fonction du filtre, toutes les couleurs à transformer sont affectées par toutes les régions. Plus une couleur est proche du centre d'une région, plus elle est affectée par cette région. Les couleurs qui sont dans aucune des boîtes des régions ne sont pas modifiées.

Résultats

Comme premier résultat de l'application locale, on a repris l'exemple de la *compression* avec deux régions complètement disjointes à la figure 5.15. Chaque région de l'image de départ est associée à la région de l'image objectif ayant la même position. La sélection des couleurs à partir de ses régions a été faite par la méthode vectorielle. Les transformations de couleurs sont des transformations linéaires en RGB. Puisque c'est une application locale, une partie du ciel et une partie de la mer n'ont pas été transformées. Ces couleurs sont simplement hors des boîtes de couleurs associées aux régions. Étant donné que les boîtes sont disjointes, une transformation affecte les couleurs du ciel qui sont à l'intérieur de la boîte et une transformation différente agit sur celles de la mer. Aucune couleur utilise les deux transformations. Si on calcule l'erreur sur toute l'image, on obtient 6.77 en moyenne et 84.4 comme variance. Ces valeurs sont grandes parce que l'application locale ne transforme pas toutes les couleurs de l'image. En calculant l'erreur sur le rectangle apparaissant sur l'image résultat, on obtient une erreur moyenne de 1.71 et une variance de 0.825.

La figure 5.16 montre le résultat de l'application locale avec trois régions non-disjointes. Encore une fois, on utilise la méthode vectorielle et la transformation linéaire en RGB. Les régions sélectionnées sont indiquées sur les images de départ et objectif par des rectangles. Les régions sont associées à celles qui ont la même position dans l'autre image. Puisque certaines couleurs font partie de plus d'une boîte, on utilise la pondération de l'équation 5.6. La région du rocher donne une boîte très grande parce que les boîtes sont alignées sur les axes R, G et B et que les couleurs de cette région sont alignées selon l'axe des gris (diagonale allant du noir au blanc). Cette grande boîte inclut même les couleurs des palmiers et, par conséquent, ces couleurs sont transformées. Les régions étant non-disjointes, les couleurs incluses dans les boîtes sont affectées par la transformation de chaque région. Les transformations de chaque région ne doivent pas affecter trop de couleurs à l'extérieur du cube RGB qu'elles

soient à l'intérieur ou hors de la boîte de la région. Chaque transformation qui donne une couleur à l'extérieur du cube RGB entraîne la couleur finale vers une solution inaffichable. L'influence des transformations de chaque région à l'extérieur de leur boîte peut être réduite en augmentant le n de l'équation 5.4. Cependant, il ne faut pas oublier que l'augmentation de n crée un plateau dans la fonction $H(r, g, b)$. Ce plateau crée une discontinuité entre les couleurs faisant partie d'une boîte et celles à l'intérieur de plusieurs boîtes. La couleur incluse dans une seule boîte sera influencée pratiquement par une seule transformation, tandis que les autres seront influencées à part égale par les transformations des boîtes incluant ces couleurs. Pour la figure 5.16, n égale deux. L'erreur moyenne est de 5.32 et la variance de 63.9 lorsque le calcul se fait sur toute l'image. En calculant l'erreur sur la même région que celle utilisée à la figure 5.15, on obtient 1.69 pour l'erreur moyenne et 0.785 pour la variance.

5.2.2 Application globale

L'application globale transforme tous les pixels de l'image peu importe si la couleur est à l'intérieur ou non d'une boîte d'une région. S'il y a une seule région, on transforme toutes les couleurs de l'image selon la transformation calculée pour cette région.

S'il y a plusieurs régions, on se sert de la pondération de l'équation 5.6 pour transformer toutes les couleurs de l'image. Encore une fois, toutes les régions affectent la transformation d'une couleur.

Résultats

Pour l'application globale, on utilise l'exemple de la *compression*. La figure 5.17 montre le résultat de l'application globale avec cinq régions dans les images de départ et objectif. Ces régions sont non-disjointes et elles utilisent la méthode vectorielle pour les associations de couleurs avec une transformation linéaire en RGB. Avec l'application globale, tous les pixels de l'image sont affectés par la transformation de chaque région selon l'équation de pondération 5.6. Dans ce cas, n est égal à deux. L'erreur est calculée sur toute l'image. L'erreur moyenne est de 3.84 et la variance de 13.3 comparativement à 2.77 en moyenne et 9.66 pour la variance avec la région de la figure 5.9 et la méthode vectorielle sur toute l'image.



FIG. 5.15: Correspondance de plusieurs régions disjointes avec l'application locale. En haut : l'image de départ avec les régions sélectionnées. À gauche : l'image objectif avec les régions choisies. À droite : le résultat obtenu avec la région sur laquelle est calculée l'erreur.



FIG. 5.16: Correspondance de plusieurs régions non-disjointes avec l'application locale. En haut : l'image de départ avec les régions sélectionnées. À gauche : l'image objectif avec les régions choisies. À droite : le résultat obtenu.

Dans ce chapitre, nous avons vu deux algorithmes pour associer automatiquement des couleurs d'une région aux couleurs d'une autre région : la méthode statistique et la méthode vectorielle. Ces deux méthodes ont leurs forces et leurs faiblesses. Les deux méthodes ont besoin que les regroupements de couleurs soient dans des proportions similaires dans les deux régions. Avec la méthode vectorielle, les régions sélectionnées peuvent avoir un nombre de regroupements de couleurs illimité. Cette méthode retrouve très bien les rotations, les translations et les changements d'échelle positif en RGB, mais elle ne peut pas retrouver les cisaillements en RGB. La méthode statistique ne retrouve pas toujours les rotations en RGB, mais retrouve les autres transformations. Cependant, avec la méthode statistique, s'il y a trop de regroupements de couleurs dans les régions, certaines aberrations peuvent apparaître dans les résultats. S'il y a un seul groupe de couleurs formant approximativement une droite, la méthode statistique risque de créer des artefacts. Ces deux méthodes ne sont pas les seules façons de sélectionner les couleurs d'une région. On a essayé un algorithme génétique dans certaines situations bien précises, mais c'est nettement plus long et on perd l'interactivité. On pourrait utiliser l'approche de compression d'histogrammes proposée par Larson et al. [LRP97] décrite au chapitre 2 pour transformer l'histogramme de l'image de départ afin qu'il ressemble à celui de l'image objectif.

La seconde partie traite de la gestion de l'association de plusieurs régions. La fonction de pondération utilisée n'est pas la seule qui pourrait être utilisée. Mais elle permet de pondérer en fonction de la densité des boîtes et de la distance entre la couleur et le centre des boîtes. Elle ne crée pas de discontinuité entre les couleurs incluses dans une seule boîte et celles incluses dans plusieurs boîtes si n n'est pas trop grand. Cette pondération n'est pas assujettie à la position des pixels dans l'image mais seulement à la position de la couleur du pixel dans l'espace RGB.

Ces méthodes de mise en correspondance de régions ou de couleurs utilisent les données des images de départ et objectif. Ces images sont souvent acquises par un appareil qui capte un certain intervalle de couleurs. Les couleurs à l'extérieur de cet intervalle sont encodées comme la couleur maximale enregistrée par l'appareil. Ceci crée des taches de couleur identique alors que les couleurs correspondantes dans la scène ne sont pas identiques. Dans le chapitre suivant, on développera une méthode pour extrapoler les couleurs saturées d'une image.



FIG. 5.17: Correspondance de plusieurs régions non-disjointes avec l'application globale. En haut : l'image de départ avec les régions sélectionnées. À gauche : l'image objectif avec les régions choisies. À droite : le résultat obtenu.

Chapitre 6

Correction de couleurs saturées

Une caméra est sensible à l'intérieur d'un certain intervalle de couleurs. Cet intervalle est limité par la technologie des senseurs ou des produits sur la pellicule photographique. Donc, si l'on photographie une scène et que des couleurs de cette scène sont hors de l'intervalle de la caméra, la caméra "sature" ces couleurs, c'est-à-dire qu'elle les enregistre comme si elles étaient de la même couleur que la couleur maximale que la caméra peut capter dans cette dimension. De telles scènes ne sont pas rares. Il suffit qu'il y ait des sources de lumière directement visibles ou des objets spéculaires qui réfléchissent fortement la lumière.

Plusieurs caméras digitales captent les couleurs par trois canaux séparés : un pour chaque composante : rouge, vert et bleu. Pour ces caméras, une couleur située à l'extérieur de l'intervalle pour un ou plusieurs canaux est ramenée à la borne de l'intervalle la plus proche selon chaque canal saturé, sinon elle est enregistrée telle quelle par les canaux non débordés. En plus de modifier la luminance originale de la couleur, cette façon de tronquer la couleur selon les axes R, G et B change aussi la teinte de la couleur. Par exemple, la couleur (300, 200, 200) est captée comme (255, 200, 200), ce qui change la proportion de rouge par rapport aux autres composantes. Donc, on n'obtient ni la vraie teinte ni la vraie luminance des couleurs saturées.

Ces couleurs saturées peuvent occasionner des erreurs dans le résultat final. Prenons une image dont une partie des couleurs sont tronquées. Supposons qu'il existe, avant la troncature, une transformation uniforme sur toute l'image permettant de la transformer pour obtenir exactement le résultat voulu. Une fois les couleurs tronquées, la transformation ne donnera plus le résultat voulu. Chaque pixel où la couleur a été

tronquée donnera une erreur. Si on utilise les couleurs saturées parmi les associations, on obtiendra une autre transformation que celle recherchée. Cette nouvelle transformation donnera peut-être des erreurs moins grandes dans la zone des couleurs saturées, mais provoquera des erreurs sur tout le reste de l'image. Il faudrait pouvoir utiliser les couleurs non tronquées.

Pour obtenir les vraies couleurs d'une scène, Debevec et Malik [DM97] prennent une série de photos de la même scène avec des temps d'exposition différents. Après une analyse de la réaction du film à l'exposition, ils obtiennent une carte de radiance de la scène. De cette façon, ils obtiennent les "vraies" couleurs en fonction de la réponse du film, quelle que soit la spécularité des objets de la scène.

Cependant, on ne peut pas toujours avoir une série de photos de la même scène avec différents temps d'exposition. Les images peuvent être des photos prises par quelqu'un d'autre. Les images peuvent avoir été créées synthétiquement par une autre personne et, encore là, les données complètes peuvent ne pas être accessibles. Les scènes à photographe peuvent avoir des objets en mouvement et il est impossible d'avoir exactement la même scène d'une image à l'autre. Dans tous ces cas, il est impossible d'avoir la carte de radiance de la scène et de ce fait, les "vraies" couleurs de la scène. Les seules données disponibles sont alors les couleurs enregistrées dans l'image.

Dans ce chapitre, une technique sera présentée pour extrapoler des couleurs plus appropriées pour remplacer les couleurs saturées. Elle est basée sur des couleurs non saturées faisant partie du même objet. Cette technique cherche une approximation qui est plus près de la couleur véritable que la couleur saturée. Elle utilise le modèle de réflexion dichromatique introduit par Shafer [Sha85] et utilisé par plusieurs [KSK87] [Ger87] [NS92] [Kli93].

La première section est consacrée au modèle de réflexion dichromatique. Elle est suivie par son application dans l'extrapolation des couleurs saturées. La dernière section présente des exemples d'application de la technique d'extrapolation des couleurs saturées.

6.1 Modèle de réflexion dichromatique

Le modèle de réflexion dichromatique cherche à représenter les couleurs réfléchies par les objets diélectriques non-uniformes et opaques. Les objets diélectriques sont des objets

qui ne conduisent pas l'électricité contrairement aux métaux. Non-uniforme signifie que l'objet est non homogène, c'est-à-dire qu'il contient des pigments plus ou moins bien répartis dans la surface de l'objet. Dans cette catégorie, on regroupe les objets plastiques, la céramique, la peinture, le papier et bien d'autres. Le modèle ne s'applique pas aux objets métalliques, transparents ou sans pigment à l'intérieur de la surface de l'objet.

Shafer [Sha85] sépare la réflexion de la lumière sur un objet en deux phénomènes : la réflexion de surface et la réflexion interne (*body reflection*). La figure 6.1 illustre les deux phénomènes. La réflexion de surface ne pénètre pas à l'intérieur de l'objet ; elle représente la réflexion parfaite à la surface de l'objet. Cette réflexion peut être dans une seule direction ou très dispersée dépendamment de la rugosité de la surface. Plusieurs modèles de réflexion de surface ont été introduits en infographie [BT75] [Bli77] [War92] [HTSG91].

Dans la réflexion interne, la lumière pénètre dans l'objet et frappe des particules de couleur (pigments) qui absorbent une partie de la lumière et réfléchissent le reste. Après possiblement plusieurs réflexions, la lumière est réfléchie à l'extérieur de l'objet. C'est la réflexion interne.

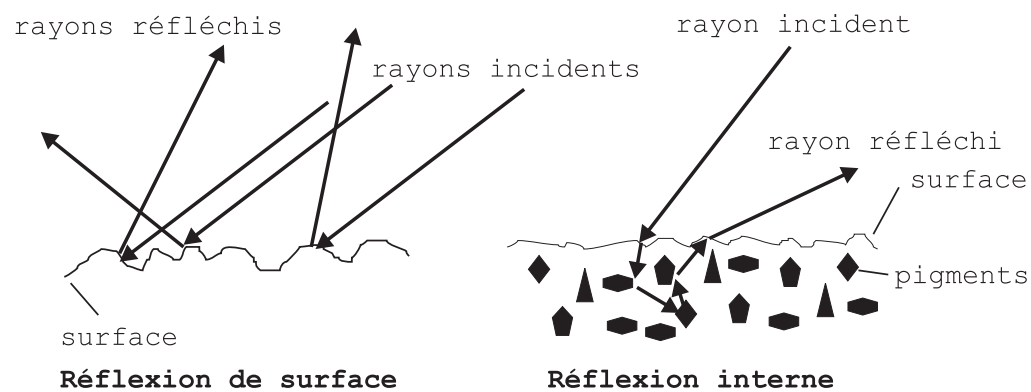


FIG. 6.1: Réflexions de surface et interne.

Les deux types de réflexion sont présents dans le cas des objets diélectriques non-uniformes et opaques. Pour les objets métalliques, la réflexion interne est pratiquement inexistante : toute la lumière est réfléchie sur la surface. Les objets diélectriques uniformes sont dans le même cas. Ils n'ont pas de réflexion interne car ces objets n'ont pas de pigments pour réfléchir la lumière à l'intérieur de la surface de l'objet. Pour les objets transparents, un nouveau phénomène s'ajoute : la lumière transmise par

l'objet, c'est-à-dire la lumière qui traverse l'objet. Pour tous ces objets, le modèle de réflexion dichromatique ne s'applique pas puisqu'ils n'ont pas la réflexion de surface et la réflexion interne sans la transmission.

Le modèle de réflexion dichromatique est illustré par l'équation suivante :

$$L(\lambda, i, e, g) = m_s(i, e, g)c_s(\lambda) + m_b(i, e, g)c_b(\lambda) \quad (6.1)$$

où λ est la longueur d'onde de la lumière, i l'angle d'incidence de la lumière par rapport à la normale de la surface N , e l'angle entre le point de vue et la normale, et g l'angle entre la lumière et le point de vue. L est l'intensité de la lumière réfléchiée. La figure 6.2 montre les différents angles. Shafer sépare chaque type de réflexion en deux termes : la composition c_s (c_b) qui est fonction uniquement de la longueur d'onde et la magnitude m_s (m_b) qui dépend des aspects géométriques de la scène. Les termes de composition sont des vecteurs dans l'espace RGB représentant la couleur de la réflexion. Pour cette raison, c_s et c_b sont des vecteurs couleur. Les termes de magnitude sont des scalaires qui modifient l'intensité de la couleur de réflexion selon la géométrie. Ce modèle suppose une seule source lumineuse et aucune interr réflexion entre les surfaces. Il suppose aussi que les objets soient de couleur uniforme.

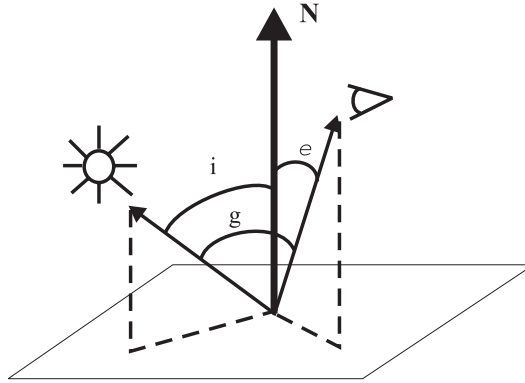


FIG. 6.2: Aspects géométriques de la scène. Le vecteur N représente la normale. i est l'angle d'incidence. e est l'angle entre le point de vue et la normale. g est l'angle entre la source de lumière et le point de vue.

Puisque c_s et c_b ne varient pas en fonction de la géométrie pour un objet de couleur uniforme, les vecteurs couleur c_s et c_b sont les mêmes sur toute la surface de l'objet. Seuls les scalaires m_s et m_b varient. Pour un objet d'une couleur uniforme, les couleurs d'une surface diélectrique forment un plan dans l'espace RGB. Ce plan comprend toutes les

combinaisons linéaires des vecteurs couleur c_s et c_b . Un exemple de combinaison linéaire est présenté à la figure 6.3.

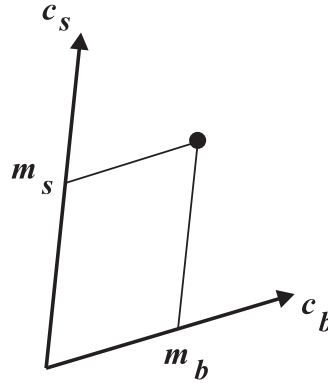


FIG. 6.3: Combinaison linéaire de vecteurs couleur.

Klinker et al. [KSK87] et Gershon [Ger87] ont observé que les couleurs d'un objet de couleur uniforme ne sont pas également distribuées sur la surface du plan. Les couleurs forment plutôt deux groupements linéaires constituant une sorte de "T". La figure 6.4 montre un exemple de deux groupements linéaires. Le premier groupe comprend les pixels sujets à une réflexion interne prédominante. Les couleurs de ce groupe sont mates. Ce groupe sera appelé le vecteur mat. Ce vecteur part de la couleur de la région de l'objet la moins exposée à la lumière et se dirige vers la "couleur de l'objet". Si l'objet a une partie dans l'ombre, le point de départ du vecteur mat sera le point le plus près du coin noir du cube RGB. Le second vecteur regroupe les pixels dont la réflexion de surface est plus importante. Ce groupe comprend les "highlights" de l'objet, on l'appellera le vecteur "highlight". Le vecteur "highlight" part d'un point du vecteur mat et se dirige vers la "couleur de la lumière". Le point de départ du vecteur "highlight" dépend de l'intensité de la réflexion interne aux pixels faisant partie du "highlight" par rapport à l'intensité maximum enregistrable. Pour un objet diélectrique, non-uniforme, opaque et d'une seule couleur, les variations de la couleur apparaissant sur l'objet illuminé forment un vecteur mat et un vecteur "highlight" sauf si le point de vue ne permet pas de voir le "highlight". Pour un objet métallique, puisqu'il n'y a pas de réflexion interne, les couleurs réfléchies par l'objet forment un seul vecteur dans l'espace RGB.

Novak et Shafer [NS92] utilisent ces informations pour approximer certaines propriétés d'une scène telles l'intensité de la lumière, la rugosité de la surface et l'angle entre la source lumineuse et la caméra. Par la suite, ils peuvent déterminer avec plus

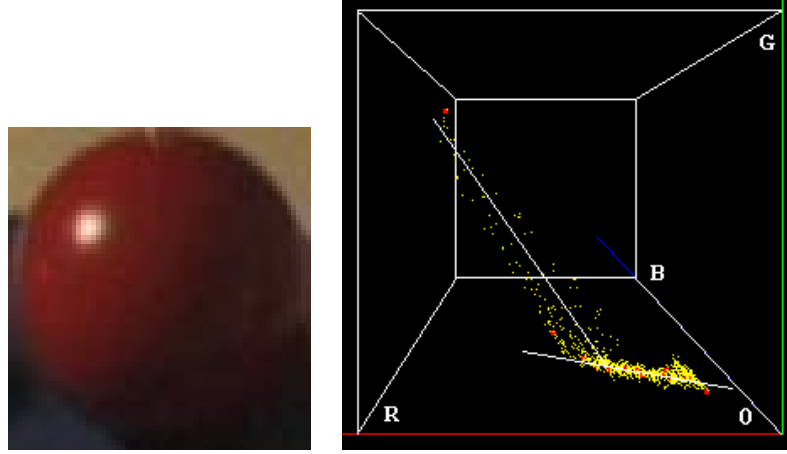


FIG. 6.4: Groupements linéaires en forme de T. À gauche : image d’une sphère rouge. À droite : distribution des couleurs de la sphère dans l’espace RGB. Le groupement partant du point le plus près du coin noir (coin 0) est le vecteur mat. Le second groupement est le vecteur “highlight”.

de précision la couleur de la source lumineuse.

Klinker [Kli93] utilise le modèle de réflexion dichromatique pour séparer les objets d’une scène. Il peut seulement séparer des objets diélectriques non-uniformes et opaques parce qu’il lui faut deux groupes de couleurs pour chaque objet. Il est donc très important d’orienter la caméra afin que les “highlights” des objets soient visibles.

Que les objets soient métalliques ou diélectriques, non-uniformes et opaques, ils ont tous un ou deux vecteurs couleur s’ils ont une couleur uniforme. Nous utiliserons cette propriété pour notre application.

6.2 Extrapolation des couleurs saturées

Pour notre méthode d’extrapolation des couleurs saturées d’une image, on exploitera le groupement des couleurs en vecteurs couleur pour un objet d’une couleur uniforme. Il faut d’abord calculer le vecteur couleur qui passe par les “vraies” couleurs des pixels saturés. Les composantes non saturées du pixel serviront alors pour projeter la couleur saturée sur le vecteur couleur. Par exemple, soient la composante r saturée dans la couleur (r, g, b) , le vecteur (v_r, v_g, v_b) et un point du vecteur (p_r, p_g, p_b) . La figure 6.5 montre cet exemple en deux dimensions pour faciliter la compréhension. L’équation permettant de projeter la couleur saturée sur le vecteur est :

$$\begin{aligned}
t &= \frac{g - p_g}{v_g} \\
r' &= v_r * t + p_r.
\end{aligned}
\tag{6.2}$$

Pour cette équation, on a utilisé la composante g pour trouver la nouvelle valeur r' , bien que b aurait pu être utilisé. L'important est d'utiliser une composante non saturée. Évidemment, si les trois composantes sont saturées, on ne peut pas calculer une approximation de la couleur véritable par manque d'information. Lorsque deux composantes sont saturées, on n'a pas le choix, il faut utiliser la troisième. En théorie, le choix de la composante pour l'extrapolation lorsqu'une seule composante est saturée n'est pas important puisque les deux autres composantes devraient donner la même valeur. En pratique, les caméras ne sont pas parfaites. Elles enregistrent du bruit. Avec ce bruit, il est peu probable que les deux composantes donnent les mêmes valeurs. Pour cette raison, on calcule la valeur de t à partir des deux composantes et on fait la moyenne des deux valeurs obtenues. On utilise le t moyen dans l'équation 6.2. La nouvelle couleur n'est donc qu'une approximation plus juste que la couleur saturée. Au moins, la nouvelle couleur aura plus de chance d'avoir la bonne teinte.

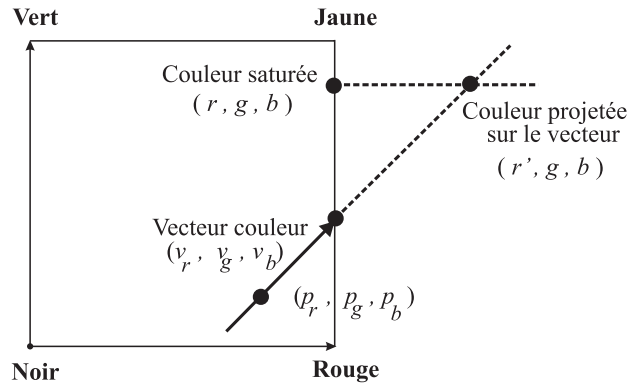


FIG. 6.5: Exemple de l'extrapolation d'une couleur saturée en deux dimensions.

Cette technique d'extrapolation est basée sur le vecteur couleur. Pour l'obtenir, on demande à l'utilisateur de sélectionner une région contenant les pixels saturés et un dégradé suffisant de pixels non saturés appartenant au même vecteur couleur. La région peut être choisie à l'aide d'un rectangle ou d'un masque. Pour le masque, l'utilisateur doit promener la souris sur tous les pixels qu'il veut. Une fois les pixels choisis, on trouve le vecteur propre ayant la plus grande valeur propre passant par les pixels non saturés de la région. Le

vecteur propre est calculé de la même façon que dans la méthode vectorielle de mise en correspondance de régions. Pour le point (p_r, p_g, p_b) par lequel passe le vecteur couleur, on utilise la couleur moyenne des pixels non saturés de la région. Les images ont un certain bruit, donc les groupes de couleurs des objets ne sont pas exactement des lignes. Ces groupes sont plutôt des nuages orientés dans une direction. Le plus important pour le choix de la région est d'éviter de prendre des pixels dont la couleur ne fait pas partie du nuage du vecteur couleur. Chaque couleur à l'extérieur du nuage fausse la direction du vecteur propre et le point moyen. Lorsqu'on veut identifier le vecteur couleur d'une réflexion interne, il est assez facile de savoir où s'arrêter dans le choix de la région. Pour un vecteur couleur d'une réflexion de surface, la situation se complexifie. Si l'objet a aussi un vecteur pour la réflexion interne, il est difficile de savoir où s'arrêtent les pixels appartenant au "highlight" et ceux appartenant au vecteur mat, surtout que souvent la plupart des pixels du "highlight" sont saturés. Il faut faire un compromis et prendre des pixels autour des pixels saturés sans aller trop loin.

Notre technique ne s'applique pas uniquement aux objets diélectriques non-uniformes et opaques. Comme on l'a expliqué plus haut, les objets métalliques ont aussi un vecteur couleur. Donc, il est possible de trouver le vecteur couleur en choisissant bien les régions. Les objets transparents peuvent poser des problèmes parce qu'on peut voir d'autres objets au travers et qu'ainsi il est très difficile de délimiter un seul vecteur couleur. Le modèle de réflexion dichromatique ne tient pas compte des interréllections entre les objets. Dans des images réalistes, l'interréllection peut jouer un rôle plus ou moins important. Lorsqu'on choisit une région pour trouver le vecteur couleur, il faut s'assurer de ne pas prendre des pixels qui incluent les couleurs des autres objets. Lorsqu'un objet réfléchit comme un miroir, les couleurs des autres objets peuvent se refléter sur lui. Les pixels sélectionnés ne doivent pas inclure les pixels de ces couleurs.

La prochaine section présente quelques exemples.

6.3 Exemples

Le premier exemple est une scène synthétique qu'on peut voir à la figure 6.6. Cet exemple sera appelé *dégradé*. Les images de départ et objectif sont formées de bandes verticales de couleurs uniformes. Les différentes couleurs sont toutes disposées sur un vecteur dans l'espace RGB et elles sont espacées également en RGB. Dans l'image de

départ, les couleurs sont plus espacées que dans l'image objectif. Les couleurs de la partie droite de l'image de départ se retrouveraient hors du cube RGB et sont donc tronquées selon les axes sur le cube pour être affichables. Ces couleurs ont donc une mauvaise proportion entre la composante saturée et les autres composantes. Dans l'image objectif, les couleurs sont moins espacées et elles ne sortent pas du cube. La figure 6.6 présente les images de cet exemple. L'extrapolation déplace les couleurs saturées à l'extérieur du cube RGB sur le vecteur couleur. Pour afficher l'image extrapolée, on projette les couleurs extérieures sur le cube mais en maintenant la couleur sur le vecteur couleur. À l'affichage, les couleurs qui étaient à l'extérieur du cube ont donc toutes la même couleur. Cependant, pour la transformation, on n'utilise pas les couleurs affichées mais bien les couleurs à l'extérieur du cube. De cette manière, on obtient le bon dégradé après la transformation. Dans les deux transformations, on a utilisé la même série de correspondances choisies par l'utilisateur. Si l'image de départ n'était pas saturée, il existerait une transformation linéaire en RGB permettant de passer de l'image de départ à l'objectif. On utilise une transformation linéaire en RGB.

Au tableau 6.1, on retrouve l'erreur moyenne, la variance et l'erreur maximale en LUV entre l'image objectif et les images de départ, transformées sans et avec extrapolation. Les statistiques sont calculées sur toute l'image.

Image	Erreur moy LUV	Variance de l'erreur LUV	Erreur max LUV
a) départ et b) objectif	15.30	23.3	25.17
b) objectif et c) sans	2.40	13.5	12.43
b) objectif et e) avec	0.328	0.0518	1.15

TAB. 6.1: Résultats des transformations avec et sans extrapolation sur les images de la figure 6.6 : *dégradé*. Les statistiques sont calculées sur toute l'image.

Les trois exemples suivants sont tirés d'une série de photos prises par Debevec pour construire une carte de radiance [DM97]. Ces photos sont prises avec différents temps d'exposition. Avec ces différents temps d'exposition, certaines zones sur certaines photos sont saturées alors que sur d'autres photos ces zones ne sont pas saturées. Le premier exemple sera appelé *l'oeuf* et est présenté à la figure 6.7. La région saturée qu'on a choisie est une région où la réflexion interne est prédominante. Une région mate est rarement saturée. Pour qu'elle le soit, il faut que le temps d'exposition soit trop long pour cette partie de la photo. Pour trouver le vecteur couleur de l'oeuf, on a choisi la

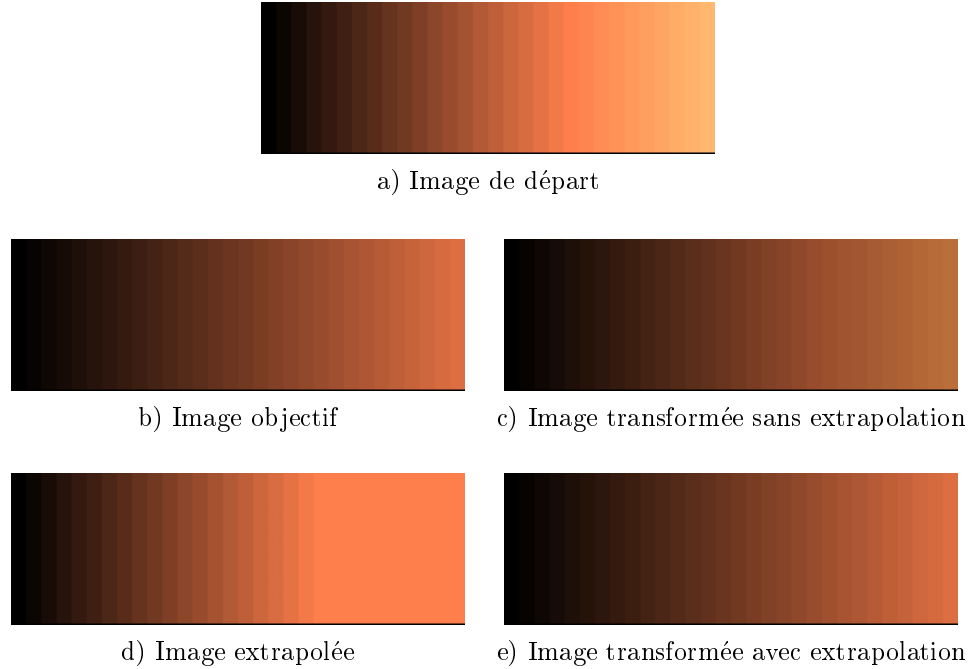


FIG. 6.6: Correction des couleurs saturées sur le *dégradé*. En haut : image de départ. Au centre à gauche : image objectif. Au centre à droite : image transformée sans extrapolation préalable. En bas à gauche : image extrapolée avec couleurs extérieures projetées le long du vecteur couleur. En bas à droite : image transformée avec une extrapolation préalable. Les correspondances ont été choisies par l'utilisateur et on a utilisé une transformation linéaire en RGB. Les images ont 300 par 100 pixels.

majorité des pixels contenant l'oeuf à l'aide d'un masque. La figure 6.7 présente les images de l'exemple de l'*oeuf*. Les correspondances de couleurs sont faites à l'aide de la méthode vectorielle. Les régions associées sont indiquées sur les images de départ et objectif. Ces rectangles ne servent pas au calcul du vecteur couleur. On a utilisé une transformation linéaire en RGB.

Les résultats ne sont pas aussi visibles que dans l'exemple du *dégradé* parce que l'extrapolation est moins importante. Mais en calculant l'erreur sur la région de l'image de départ, on obtient une amélioration. Le tableau 6.2 présente l'erreur moyenne en LUV, la variance et le maximum sur la région entre les images de la figure 6.7.

Nous reprenons les mêmes images que l'exemple de l'*oeuf* en nous concentrant cette fois-ci sur le "highlight" de la sphère rouge. La figure 6.8 montre l'exemple de la *sphère*. Les couleurs de la sphère rouge se répartissent en deux groupements : un vecteur mat et un vecteur "highlight". Pour le calcul du vecteur "highlight", nous avons sélectionné

Image	Erreur moy LUV	Variance de l'erreur LUV	Erreur max LUV
a) départ et b) objectif	13.03	0.975	16.29
b) objectif et c) sans	2.63	2.32	9.28
b) objectif et e) avec	2.59	2.74	8.51
a) départ et d) extrapolée	0.0594	0.0472	1.92

TAB. 6.2: Résultats des transformations avec et sans extrapolation sur les images de la figure 6.7 : *oeuf*. Les statistiques sont calculées sur la région de l'image de départ.

tous les pixels du “highlight” de la sphère. Les correspondances de couleurs sont faites à l'aide de la méthode statistique. Les régions associées sont indiquées sur les images de départ et objectif. Nous avons utilisé une transformation linéaire en RGB.

Dans cet exemple, le vecteur “highlight” est assez difficile à calculer puisqu'il n'y a pas beaucoup de pixels dans le “highlight”. De plus, on ne peut pas sélectionner les pixels autour du “highlight” parce qu'ils peuvent appartenir au groupement du vecteur mat. Les statistiques sur l'erreur sont présentées dans le tableau 6.3.

Image	Erreur moy LUV	Variance de l'erreur LUV	Erreur max LUV
a) départ et b) objectif	18.50	15.9	26.7
b) objectif et c) sans	8.78	78.1	70.3
b) objectif et e) avec	8.57	69.4	62.6
a) départ et d) extrapolée	0.0826	0.353	9.16

TAB. 6.3: Résultats des transformations avec et sans extrapolation sur les images de la figure 6.8 : *sphère*. Les statistiques sont calculées sur la région de l'image de départ.

Le dernier exemple est également tiré des photos de Debevec. Cet exemple sera appelé la *loupe*. Cette fois la zone ciblée est la partie saturée de l'armature métallique d'une loupe. Cette partie de la loupe étant métallique, elle réfléchit la lumière à la surface de l'objet. La région est un “highlight”. Les pixels servant au calcul du vecteur couleur sont choisis à l'aide d'un masque en évitant les reflets des autres objets. La figure 6.9 présente les images de l'exemple de la *loupe*. Les correspondances sont faites à l'aide de la méthode vectorielle et on utilise une transformation linéaire en RGB.

Encore une fois, l'extrapolation apporte peu de changements sur toute la région. Malgré tout, certaines couleurs sont déplacées de façon notable puisque la différence maximale entre les pixels est de 7.73 en LUV entre l'image de départ et celle extrapolée. L'extrapolation apporte une légère amélioration comme le montrent les statistiques du

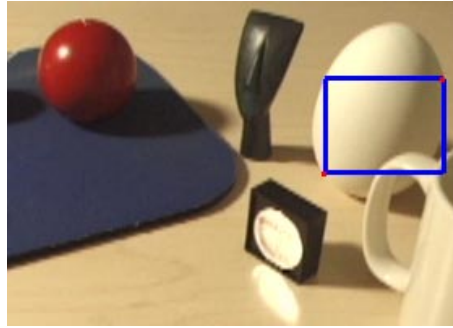
tableau 6.4. L'erreur est calculée sur la région de l'image de départ de la figure 6.9.

Image	Erreur moy LUV	Variance de l'erreur LUV	Erreur max LUV
a) départ et b) objectif	13.39	13.06	20.35
b) objectif et c) sans	5.58	5.45	17.26
b) objectif et e) avec	5.26	7.59	14.44
a) départ et d) extrapolée	0.814	3.22	7.73

TAB. 6.4: Résultats des transformations avec et sans extrapolation sur les images de la figure 6.9 : *loupe*. Les statistiques sont calculées sur la région de l'image de départ.

Dans ce chapitre, nous avons exposé une technique pour obtenir une meilleure approximation de la “vraie” couleur d’un pixel saturé. Cette technique est basée sur le fait qu’un objet d’une couleur uniforme forme un ou plusieurs nuages de points le long d’un vecteur dans l’espace RGB. Il suffit de trouver le vecteur couleur de la région entourant les pixels saturés pour positionner la couleur saturée sur ce vecteur. Cette technique fonctionne sur tous les types d’objets. Cependant, le vecteur couleur peut être difficile à cerner dans certaines situations. Il faut avoir un dégradé suffisant autour des pixels saturés. Si le changement de couleur dû à la géométrie de l’objet est trop petit, le vecteur propre calculé pourrait être différent du vecteur couleur recherché. Pour les objets qui ont à la fois des zones mates et des zones de “highlight”, il est un peu plus difficile de déterminer le vecteur couleur des “highlights”. Lors de la sélection des pixels entourant le “highlight”, il faut choisir un dégradé de couleurs non saturées suffisant sans sélectionner les pixels qui appartiennent à une zone mate. La frontière entre les zones mates et les zones de “highlight” est souvent difficile à déterminer. Notre technique permet de trouver une approximation seulement pour les couleurs qui ont au moins une composante non saturée. Lorsque les trois composantes sont saturées, nous n’avons pas assez d’information pour extrapoler la couleur.

Comme vous pouvez le constater, il reste encore des améliorations à apporter. Le chapitre suivant discutera de ces améliorations ainsi que des travaux futurs. Ce dernier chapitre conclura ce mémoire.



a) Image de départ



b) Image objectif



c) Image transformée sans extrapolation

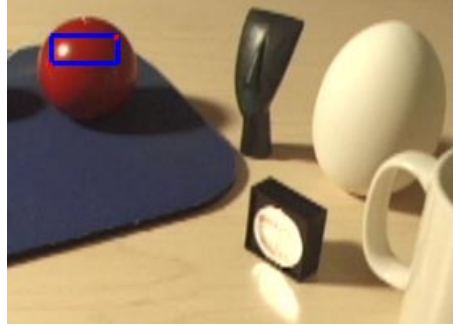


d) Image extrapolée

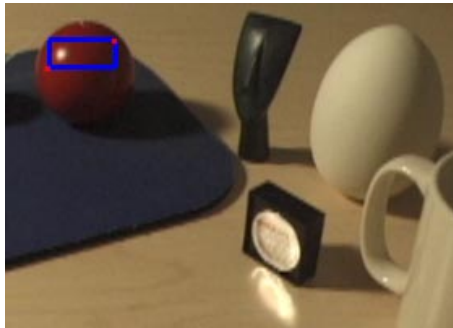


e) Image transformée avec extrapolation

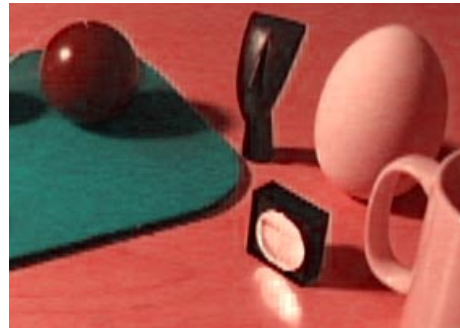
FIG. 6.7: Correction des couleurs saturées sur l'oeuf. En haut : image de départ avec la région sélectionnée. Au centre à gauche : image objectif avec la même région sélectionnée. Au centre à droite : image transformée sans extrapolation préalable. En bas à gauche : image extrapolée avec couleurs extérieures projetées le long du vecteur couleur. En bas à droite : image transformée avec une extrapolation préalable. Les correspondances ont été choisies à l'aide de la méthode vectorielle et on a utilisé une transformation linéaire en RGB. Les images ont 280 par 200 pixels. Les images de départ et objectif ont été fournies par M. Debevec.



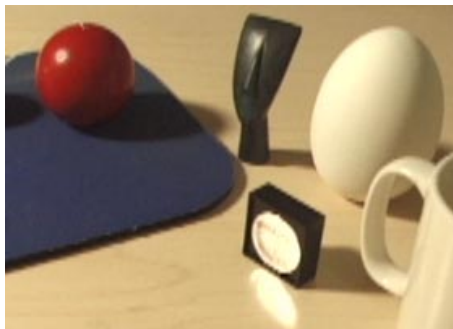
a) Image de départ



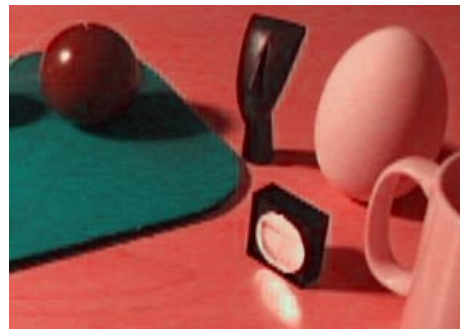
b) Image objectif



c) Image transformée sans extrapolation



d) Image extrapolée



e) Image transformée avec extrapolation

FIG. 6.8: Correction des couleurs saturées sur la *sphère*. En haut : image de départ avec la région sélectionnée. Au centre à gauche : image objectif avec la même région sélectionnée. Au centre à droite : image transformée sans extrapolation préalable. En bas à gauche : image extrapolée avec couleurs extérieures projetées le long du vecteur couleur. En bas à droite : image transformée avec une extrapolation préalable. Les correspondances ont été choisies à l'aide de la méthode statistique et on a utilisé une transformation linéaire en RGB. Les images ont 280 par 200 pixels. Les images de départ et objectif ont été fournies par M. Debevec.



a) Image de départ



b) Image objectif



c) Image transformée sans extrapolation



d) Image extrapolée



e) Image transformée avec extrapolation

FIG. 6.9: Correction des couleurs saturées sur la *loupe*. En haut : image de départ avec la région sélectionnée. Au centre à gauche : image objectif avec la même région sélectionnée. Au centre à droite : image transformée sans extrapolation préalable. En bas à gauche : image extrapolée avec couleurs extérieures projetées le long du vecteur couleur. En bas à droite : image transformée avec une extrapolation préalable. Les correspondances ont été choisies à l'aide de la méthode vectorielle et on a utilisé une transformation linéaire en RGB. Les images ont 130 par 150 pixels. Les images de départ et objectif ont été fournies par M. Debevec.

Chapitre 7

Conclusion

Ce mémoire traite de la correction des couleurs d'une image. Plusieurs éditeurs font des transformations de couleurs. Ces éditeurs fournissent habituellement une série de transformations de base qu'il faut combiner pour obtenir la transformation finale voulue. Souvent les transformations complexes obtenues à partir des transformations de base sont linéaires. Déterminer la bonne combinaison n'est pas facile parce qu'il faut choisir les transformations de base, déterminer les paramètres de ces transformations ainsi que leur ordre d'application. Nous avons développé une autre approche pour obtenir la transformation complexe. Nous associons des couleurs à changer aux couleurs désirées dans le résultat final. Ces couleurs désirées peuvent provenir d'une ou de plusieurs images ou encore d'une palette de couleurs. Leur provenance n'a aucune importance puisqu'on utilise seulement les composantes couleurs. L'utilisateur choisit un type de transformations générales telles une transformation linéaire en RGB ou en LUV ou d'un degré plus élevé. Nous calculons les paramètres de la transformation satisfaisant le mieux toutes les associations. Ensuite, nous transformons toutes les couleurs de l'image ou seulement une région avec les paramètres calculés.

Les avantages de cette méthode sont nombreux. Il est plus intuitif de choisir une série d'associations de couleurs que de déterminer la combinaison exacte des transformations de base à utiliser. Le calcul des transformations se fait de façon interactive. Qu'on ajoute une association ou qu'on change la couleur d'un membre d'une association, le résultat de la nouvelle transformation est affiché presque immédiatement. Pour les types de transformations, nous en avons exploré cinq : linéaire en RGB, quadratique en RGB, linéaire en (Luv) , linéaire en (L) et en (uv) , et linéaire en (L) et cubique en (uv) . Selon

les modifications de couleurs voulues, un type peut être plus approprié qu'un autre. Si la modification est une combinaison de rotations, translations, cisaillements et de changements d'échelle dans l'espace RGB, il est préférable d'utiliser la transformation linéaire en RGB puisque la combinaison donnera toujours une transformation linéaire en RGB. Une comparaison des cinq types de transformations a été présentée à la fin du chapitre 4. Si la transformation recherchée est du même type que la transformation employée et si les associations ne contredisent pas la transformation recherchée, en général, une augmentation des associations entraîne une amélioration du résultat.

Dans certaines situations, il est difficile pour l'utilisateur de choisir des associations sans établir de contradictions. Les images posant le plus de problèmes sont celles ayant des fréquences élevées ou, au contraire, des régions qui varient très peu. L'utilisateur peut alors éprouver des difficultés à sélectionner la couleur désirée lorsque les fréquences sont élevées. Lorsque les couleurs varient très peu, l'utilisateur a des difficultés à choisir les couleurs puisqu'il ne distingue pas très bien la variation des couleurs. Dans les deux cas, une sélection automatique des couleurs peut venir en aide à l'utilisateur. Pour la sélection automatique, l'utilisateur associe des régions de couleurs plutôt que les couleurs elles-mêmes. Nous avons développé deux méthodes pour la sélection automatique des couleurs à l'intérieur de régions : la méthode statistique et la méthode vectorielle. Ces méthodes ont chacune leurs avantages et leurs inconvénients. Selon les modifications à faire, une méthode peut être plus efficace que l'autre. Ces méthodes sont discutées au chapitre 5. À la fin de ce chapitre, nous avons présenté une façon de combiner les transformations correspondant aux associations de régions pour former une transformation. Cette transformation peut modifier toute l'image (transformation globale) ou seulement les couleurs des régions choisies (transformation locale).

Une application de ces travaux est la correction des couleurs saturées d'une image. Les caméras digitales captent un intervalle limité de luminances en fonction de la luminance générale de la scène. Certaines parties de la scène peuvent avoir une luminance hors de cet intervalle. C'est souvent le cas à l'intérieur de "highlights". Certaines caméras digitales tronquent ces couleurs selon les axes du cube RGB. En utilisant la théorie de Shafer [Sha85], nous extrapolons ces couleurs projetées vers une meilleure approximation de leur couleur réelle. Cette extrapolation peut être utilisée avant de transformer l'image pour avoir des couleurs plus appropriées dans le résultat final.

La comparaison des différentes méthodes présentées dans ce mémoire a demandé beaucoup de travail pour évaluer les résultats de ces méthodes sur différentes images. Le domaine des transformations de couleurs étant très vaste, il y a des améliorations et d'autres explorations à compléter. La section suivante expose les améliorations et les travaux futurs.

7.1 Améliorations et travaux futurs

Pour le moment, chaque association de couleurs a la même importance que les autres associations. Il n'y a pas d'association qui compte plus qu'une autre pour le calcul de la transformation. Pour donner plus de liberté à l'utilisateur, on pourrait associer un poids à chaque correspondance et ainsi permettre qu'une correspondance ait plus d'importance qu'une autre. Ceci permettrait à l'utilisateur de spécifier si les couleurs doivent être transformées obligatoirement ou de préférence vers la couleur désirée. L'association de poids peut aussi servir lors d'une transformation avec plusieurs régions associées pour favoriser une paire de régions plutôt qu'une autre. L'ajout de poids pour les associations de couleurs est relativement facile puisqu'on utilise la méthode des moindres carrés pour calculer les paramètres des transformations.

Dans le chapitre 6, pour trouver le vecteur couleur du nuage de points, on utilise toutes les couleurs désignées par la région choisie. Si la région est mal choisie, elle peut contenir des couleurs très éloignées du vecteur couleur recherché. Ces couleurs peuvent provenir d'un autre vecteur couleur, et fausseront donc les calculs du vecteur couleur. Il faudrait éliminer ces couleurs pour le calcul. Pour ce faire, on peut calculer un premier vecteur couleur du nuage de points, enlever les points distants d'un certain seuil du vecteur et recalculer le vecteur couleur. Après quelques itérations, les couleurs n'appartenant pas à ce vecteur pourraient être éliminées. Avec cette technique, les calculs pour le vecteur couleur seraient à peine plus longs, mais la direction du vecteur serait plus juste.

Pour les travaux futurs, il y a plusieurs voies à explorer. Nous en nommerons quelques unes. Au chapitre 3, nous avons présenté une méthode de comparaison entre deux images : calculer l'erreur pixel à pixel. Cette méthode est assez bonne lorsque les scènes des images sont presque identiques mais il suffit de faire une translation de la scène pour que la méthode les considère comme totalement différentes. Pour donner une

seconde indication sur la ressemblance entre deux images, on pourrait, dans un premier temps, utiliser les transformées de Fourier. Les transformées de Fourier sont invariables à la translation. Ainsi elles pourraient nous fournir une indication supplémentaire. Les transformées de Fourier ne résoudreont pas tous les problèmes. Elles serviront uniquement comme un premier pas puisque la comparaison d'images est beaucoup plus complexe qu'il n'y paraît.

Dans le chapitre 4, nous avons présenté les types de transformations explorées pour ce mémoire. Ces types de transformations sont des transformations polynômiales dans l'espace RGB ou LUV. On pourrait faire ces transformations dans d'autres espaces de couleurs comme HSV, HLS, HSI ou CIE LAB. Puisqu'il n'existe pas de transformation linéaire permettant de passer de l'espace RGB à ces espaces, ils s'adapteront peut-être mieux à certaines transformations. Il existe aussi d'autres types de transformations. Nous en avons donné un aperçu au début du chapitre 4. L'utilisation de fonctions par parties (par exemple : B-splines, Bézier) dans les transformations pourrait s'avérer une voie d'exploration profitable. Les fonctions par parties de degré relativement bas permettent d'interpoler les contraintes sans osciller comme une transformation polynômiale de degré plus élevé. Avec des fonctions par parties, chaque association aurait une zone d'influence limitée. De cette façon, une mauvaise association n'affecterait pas toutes les couleurs de l'image mais seulement une région. Ces deux caractéristiques pourraient être suffisamment avantageuses pour compenser le temps de calcul supplémentaire.

Dans le chapitre 2, nous avons discuté de l'égalisation d'histogrammes légèrement modifiée de Larson et al. [LRP97]. Cette égalisation d'histogrammes pourrait nous servir pour les associations de régions. On fait l'histogramme de chacune des régions. Au lieu d'égaliser l'histogramme, on le transforme en se rapprochant de l'histogramme de la seconde région. Comme pour l'application de Larson et al. [LRP97], il ne faut pas que la pente de la transformation devienne trop grande pour éviter des variations trop prononcées par rapport à la scène. Dans notre cas, on ne veut pas transformer uniquement la luminance, on veut modifier toutes les composantes de la couleur. Un histogramme en trois dimensions ne serait pas manipulable. Deux choix s'offrent à nous. Nous pouvons transformer uniquement la luminance par ce moyen et modifier les composantes chromatiques d'une autre façon. Nous pouvons aussi transformer par histogramme chaque composante couleur séparément. Cette dernière méthode peut

engendrer un problème de décalage des couleurs (“color shift”). Il faudrait faire plusieurs tests pour déterminer la meilleure option.

Dans le chapitre 5, lorsqu’il y a plusieurs associations de régions, chaque association est traitée séparément. La sélection des couleurs et le calcul de la transformation sont indépendants des autres régions. Lorsque chaque association de régions a sa transformation, on pondère les transformations obtenues pour transformer toute l’image ou une partie. Cependant, si l’usager recherche une transformation affectant toute l’image de la même façon, il serait plus profitable de mettre ensemble les couleurs sélectionnées dans chaque région. Ensuite, on calcule une transformation unique à partir de toutes ces associations de couleurs. Chaque association de régions contribue à la transformation unique en apportant ses associations de couleurs choisies comme auparavant. De cette façon, on obtient une transformation unique affectant toute l’image de la même manière. La pondération des transformations peut donner une transformation différente pour chaque couleur en fonction de la position de la couleur par rapport aux régions de couleurs sélectionnées.

Dans ce mémoire, nous avons développé une nouvelle approche pour corriger les couleurs d’une image. Cette approche interactive par associations de couleurs permet d’éviter de chercher la bonne combinaison de transformations de base de la méthode traditionnelle. Nous avons exploré cinq types de transformations : linéaire en RGB, quadratique en RGB, linéaire en (Luv) , linéaire en (L) et en (uv) , et linéaire en (L) et cubique en (uv) . Nous avons présenté deux méthodes pour sélectionner des couleurs dans une région : la méthode statistique et la méthode vectorielle. Enfin, nous avons développé une technique pour extrapoler les couleurs saturées. Malgré tout ce travail, les sortes de transformations sont tellement nombreuses qu’il en reste beaucoup à explorer. De même, le nombre de façons de sélectionner des couleurs dans une région est seulement limité par l’imagination.

Bibliographie

- [BF93] Richard L. Burden et J. Douglas Faires. *Numerical Analysis*. PWS-KENT Publishing Company, 5^e édition, 1993.
- [Bli77] J. F. Blinn. « Models of Light Reflection for Computer Synthesized Pictures ». In ACM SIGGRAPH, éditeur. *Computer Graphics (SIGGRAPH '77 Proceedings)*, volume 11, pages 192–198, New York, 1977.
- [BT75] Phong Bui-Tuong. « Illumination for Computer Generated Pictures ». *Communications of the ACM*, volume 18, numéro 6, pages 311–317, juin 1975.
- [CC88] Steven C. Chapra et Raymond P. Canale. *Numerical Methods for Engineers*. McGraw-Hill Publishing Compagny, 2^e édition, 1988.
- [Che95] Shenchang Eric Chen. « QuickTime VR – An Image-Based Approach to Virtual Environment Navigation ». In Robert Cook, éditeur. *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 29–38, août 1995.
- [DM97] Paul E. Debevec et Jitendra Malik. « Recovering High Dynamic Range Radiance Maps from Photographs ». In *Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 369–378, août 1997.
- [FvDFH90] James D. Foley, Andries van Dam, Steven K. Feiner et John F. Hughes. *Computer Graphics, Principles and practice*. Addison-Wesley, 2^e édition, 1990.
- [Ger87] R. Gershon. *The Use of Color in Computational Vision*. Thèse de doctorat, Université de Toronto, 1987.
- [GW92] Rafael C. Gonzalez et Richard E. Woods. *Digital Image Processing*. Addison-Wesley, 1992.

- [HRV97] B. Hill, T. Roger et F. W. Vorhagen. « Comparative Analysis of the Quantization of Color Spaces on the Basis of the CIELAB Color-Difference Formula ». *ACM Transactions on Graphics*, volume 16, numéro 2, pages 109–154, avril 1997.
- [HTSG91] Xiao D. He, Kenneth E. Torrance, François X. Sillion et Donald P. Greenberg. « A Comprehensive Physical Model for Light Reflection ». In *Computer Graphics (SIGGRAPH '91 Proceedings)*, pages 175–186, juillet 1991.
- [JW75] D. B. Judd et G. Wyszecki. *Color in Business, Science, and Industry*. John Wiley and Sons, New York, 1975.
- [Kli93] Gudrun Klinker. *A Physical Approach to Color Image Understanding*. A. K. Peters, 1993.
- [KSK87] G. J. Klinker, S. A. Shafer et T. Kanade. « Using a Color Reflection Model to Separate Highlights from Object Color ». In *IEEE International Conference on Computer Vision (ICCV) 87*, pages 145–150, Londres, juin 1987.
- [Lai87] P. Laihanen. « Color Reproduction Theory Based on the Principles of Colour Science ». In *Proceedings of the International Association of Research Institutes for the Graphic Arts Industry*, volume 19, pages 1–36, 1987.
- [LRP97] Gregory Ward Larson, Holly Rushmeier et Christine Piatko. « A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes ». <http://radsite.lbl.gov/radiance/papers>, janvier 1997.
- [Mac93] Lindsay W. MacDonald. « Gamut Mapping in Perceptual Color Space ». In *Proceedings of the IS&T/SID Color Imaging Conference : Transform & Transportability of Color*, pages 193–196, 1993.
- [MB95] Eric N. Mortensen et William A. Barrett. « Intelligent Scissors for Image Composition ». In Robert Cook, éditeur. *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 191–198, août 1995.
- [Mun76] Munsell. *Book of Color*. Munsell Color Company, Baltimore, MD, 1976.
- [Neu37] H. E. J. Neugebauer. « Die Theoretischen Grundlagen des Mehrfarbendruckes (The Theoretical Foundation for Multicolor Printing) ». *Zeitschrift fuer Wissenschaftliche Photographie*, volume 36, numéro 4, pages 73–89, 1937.

- [NS92] Carol L. Novak et Steven A. Shafer. « Estimating Scene Properties from Color Histograms ». Rapport technique CMU-CS-92-212, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, novembre 1992.
- [Ost31] W. Ostwald. *Colour Science*. Winsor & Winsor, Londres, 1931.
- [PD84] Thomas Porter et Tom Duff. « Compositing Digital Images ». In *Computer Graphics (SIGGRAPH '84 Proceedings)*, pages 253–259, juillet 1984.
- [Poy97] Charles Poynton. « Color Reproduction ». In *SIGGRAPH '97 Cours 13, Digital Color*, pages 17–26, 1997.
- [PP95] Binh Pham et Glen Pringle. « Color Correction for an Image Sequence ». *IEEE Computer Graphics and Applications*, volume 15, numéro 3, pages 38–42, mai 1995.
- [PWSS96] Joanna L. Power, Brad S. West, Eric J. Stollnitz et David H. Salesin. « Reproducing Color Images as Duotones ». In *Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 237–248, août 1996.
- [RWP⁺95] Holly Rushmeier, Gregory J. Ward, Christine Piatko, Phil Sanders et Bert Rust. « Comparing Real and Synthetic Images : Some Ideas about Metrics ». In *Proceedings of the Eurographics Workshop 95*, pages 82–91, juin 1995.
- [SACM96] Michael Stokes, Matthew Anderson, Srinivasan Chandrasekar et Ricardo Motta. « A Standard Default Color Space for the Internet – sRGB ». [http ://www.w3.org/Graphics/Color/sRGB](http://www.w3.org/Graphics/Color/sRGB), novembre 1996.
- [SB96] Alvy Ray Smith et James F. Blinn. « Blue Screen Matting ». In *Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 259–268, août 1996.
- [SCB88] Maureen C. Stone, William B. Cowan et John C. Beatty. « Color Gamut Mapping and the Printing of Digital Color Images ». *ACM Transactions on Graphics*, volume 7, numéro 4, pages 249–292, octobre 1988.
- [Sha85] Steven A. Shafer. « Using Color to Separate Reflection Components ». *Color Research and Application*, volume 10, numéro 4, pages 210–218, Hiver 1985.

- [SOS98] Eric J. Stollnitz, Victor Ostromoukhov et David H. Salesin. « Reproducing Color Images Using Custom Inks ». In *Computer Graphics (SIGGRAPH '98 Proceedings)*, pages 267–274, juillet 1998.
- [Sub95] Gérard Subsol. *Constrution automatique d'atlas anatomiques morphométriques à partir d'images médicales tridimensionnelles*. Thèse de doctorat, L'École Centrale de Paris, 1995.
- [SW91] Maureen C. Stone et William E. Wallace. « Gamut Mapping Computer Generated Imagery ». In *Graphics Interface '91*, pages 32–39, 1991.
- [War92] Gregory J. Ward. « Measuring and Modeling Anisotropic Reflection ». In *Computer Graphics (SIGGRAPH '92 Proceedings)*, pages 265–272, juillet 1992.
- [WS82] G. Wyszecki et W. Stiles. *Color Science : Concepts and Methods, Quantitative Data and Formulae*. Wiley, New York, 2^e édition, 1982.

