Rapport de stage de Master.

Méthodes non-locales et application en colorisation d'images.

Fabien PIERRE.
TdSI, Master 2.
fabien.pierre@etu.u-bordeaux1.fr

sous la direction de
Jean-François AUJOL (IMB),
Aurélie BUGEAU (LaBRI),
Vinh-Thong TA (LaBRI).

Université Bordeaux 1.

Et soudain, il y eut avec l'Ange une troupe céleste innombrable, qui louait Dieu en disant : « Gloire à Dieu au plus haut des cieux, et paix sur la terre aux hommes de bonne volonté. »

Luc 2, 13-14.

Résumé.

Dans ce rapport nous étudions le problème de la colorisation dite basée-exemple. Il s'agit d'une méthode prenant en entrée une image en niveaux de gris et une image en couleur. Puis, elle retourne une image en couleur dont la luminance est égale à l'image en niveaux de gris, et qui est visuellement cohérente. Le modèle de référence tout au long de cet écrit a été introduit par Bugeau *et al.* dans [1]. Dans tout ce qui suit nous considérons celui-ci comme méthode de départ, auquel nous nous comparons. De plus, nous proposons une amélioration. Nous transformons le modèle initial qui travaille avec le modèle de couleur YUV par un modèle travaillant en RGB. Le modèle initial introduisait un défaut au niveau de la fonctionnelle qui rendait les images ternes et dégradait les contours. Nous corrigeons cette composante en modifiant le modèle au niveau du terme de régularisation grâce à des techniques d'analyse convexe et de géométrie de la mesure. Nous donnons un algorithme permettant de calculer efficacement une solution de ce modèle et proposons diverses améliorations quant à son initialisation. Enfin, nous proposons un critère de qualité pour juger quantitativement d'une méthode de colorisation basée-exemple.

(a) Image couleur, source d'information.

(b) Image en niveaux de gris à coloriser.

(c) Image colorisée par notre méthode.

FIGURE 1 – Exemple de colorisation basée-exemple.

Présentation de l'IMB et du LaBRI.

L'IMB est un établissement régit par contrat administratif type « Unité mixte de recherche »(référencé UMR 5251) dirigé par Jean-François Jaulent. Il regroupe les organismes suivants :

- CNRS ;
- Université Bordeaux 1 ;
- Université Bordeaux Segalen ;
- Institut Polytechnique de Bordeaux (IPB).

L'IMB est partenaire de l'INRIA et du CEA au sein des équipes-projets ALEA, CARMEN, CQFD, LFANT, MC2 et en particulier deux REALOPT et BACCHUS, que nous expliciterons ci-après. Il rassemble l'ensemble de la recherche en mathématiques du PRES bordelais, qui est un regroupement visant à améliorer la cohérence et la mutualisation des ressources disponibles pour la recherche. Les membres de l'IMB sont affectés à l'UFR Mathématiques-Informatique, à l'ENSEIRB-MATMECA, à l'UFR Sciences et Modélisation de l'Université Bordeaux Segalen, ou à l'Université Montesquieu-Bordeaux IV. Il s'agit donc d'un établissement très ouvert, dans lequel cohabitent sereinement les mathématiques fondamentales et appliquées. L'AERES (Agence d'Évaluation de la Recherche et de l'Enseignement Supérieur,) autorité administrative indépendante française chargée de l'évaluation de l'enseignement supérieur et de la recherche publique, lui a attribué un rapport élogieux en mai 2010 dont on citera quelques points intéressants :

> Le niveau de la recherche est excellent, dans un laboratoire dont la qualité est reconnue internationalement.
>
> La dynamique unitaire semble être forte. Elle se traduit par des volontés de collaboration entre équipes qu'il convient d'encourager.
>
> La façon dont l'IMB a su saisir l'opportunité offerte par l'implantation d'un centre INRIA à Bordeaux, pour développer des thèmes qui étaient jusque là à la traîne, et dynamiser plusieurs projets de recherche, est remarquable. [...] Le rayonnement national et international est indiscutable. L'intégration dans l'environnement bordelais est très bonne. L'interaction avec le LaBRI est sérieuse, mais pourrait peut-être être encore améliorée.

Ce rapport conseille également de développer des collaborations avec le LaBRI. Ce stage s'inscrit donc parfaitement dans cette volonté d'élargir les compétences et les champs d'applications des mathématiques appliquées. De plus ce stage se place en bonne position vis-à-vis d'un conseil donné par les auteurs du rapport :

> [...] rendre visible la thématique signal.

L'IMB participe actuellement à une dizaine de projets ANR, et à 4 projets du Conseil Régional d'Aquitaine (CRA).

Pour ce qui est de l'organisation générale de la recherche, elle est structurée autour de huit équipes :

- L'équipe théorie des nombres ;
- L'équipe géométrie ;
- L'équipe analyse ;
- L'équipe EDP et physique mathématique ;
- L'équipe probabilités et statistique ;
- L'équipe calcul scientifique et modélisation ;
- L'équipe mathématiques du vivant ;
- L'équipe recherche opérationnelle.

La thématique signal est à cheval sur l'équipe Analyse et celle de Calcul scientifique et Modélisation.

L'IMB est humainement riche de 152 chercheurs et enseignants-chercheurs, 22 personnels d'assistance à la recherche et 90 doctorants et post-doctorants.

Deux de mes encadrants sont maîtres de conférence au LaBRI (*Lab*oratoire Bordelais de Recherche en Informatique). Bien que mon stage ne s'y déroulait pas, j'ai eu l'occasion de m'y rendre pour quelques heures lors de réunions de travail et de conférences du groupe de travail image, commun à l'IMB et au LaBRI. C'est l'occasion de rappeler quelques

faits et jugements sur ce laboratoire. Le LaBRI est une unité de recherche associée au CNRS (UMR 5800), à l'Université Bordeaux 1 à l'IPB et à l'université Bordeaux Segalen. Comme l'IMB, il est partenaire de l'INRIA et les trois laboratoires partagent deux projets communs : BACCHUS traitant de la validation de modèles numériques basés EDP, et REALOPT dont l'objectif est de travailler sur la qualité des formulations de problèmes d'optimisation combinatoire.

En octobre 2012, il compte près de 320 personnes, dont 108 enseignants chercheurs, 39 chercheurs, 25 personnels administratifs et techniques et plus de 146 doctorants, post-doctorants et ingénieurs contractuels. Les missions du LaBRI s'articulent autour de trois axes principaux : recherche (théorique et appliquée), valorisation - transfert de technologie et formation. Le laboratoire s'articule autour de six équipes thématiques alliant recherche fondamentale, recherche appliquée et transfert technologique :

- Combinatoire et algorithmique ;
- Image et son, auquel sont rattachés mes encadrants ;
- Méthodes formelles
- Modèles et algorithmes pour la bioinformatique et la visualisation d'informations ;
- Programmation, réseaux et systèmes ;
- Supports et algorithmes pour les applications numériques hautes performances ;

L'AERES est également favorable à ce laboratoire dans son rapport :

> Très grande qualité des travaux réalisés par le laboratoire
> Rayonnement et attractivité remarquables
> Bonne intégration recherche et enseignement
> Soutien affirmé des partenaires de l'UMR
> Gouvernance intelligente et active

Confirmant ce qui était mentionné auparavant au sujet de l'IMB, ce stage s'inscrit dans une volonté du LaBRI de mener des projets scientifiques avec d'autres laboratoires bordelais, qui est spécialement bien vue par L'AERES :

> Le comité a particulièrement apprécié l'effort déployé par la direction pour intégrer l'activité du laboratoire dans son environnement scientifique immédiat (IMS, IMB, ...) et son action pour faciliter l'émergence scientifique à travers des projets transversaux.

Finalement tout ces faits, montrent que j'ai fait mon stage dans un environnement de qualité, et qu'il a permis de rapprocher deux laboratoires conformément aux indications que préconisent les rapports de l'ARES. Ce qui donne donc une utilité à ce stage par rapport à l'unité de l'Université Bordeaux 1, en plus d'avoir participer à ma formation.

Remerciements.

Je tiens à remercier Jean-François Aujol, à la fois pour ce sujet de stage mais aussi pour ces deux dernières années qui m'ont permis de poser les fondements d'un projet professionnel et trouver de la motivation dans mon étude des mathématiques. Je remercie aussi Vinh-Thong Ta et Aurélie Bugeau pour leur encadrement qui m'a permis d'avoir une vision du traitement d'image très différente de celle que j'avais auparavant. Cette diversification est très enrichissante.

Je remercie aussi Romain Yildizoglu et Nicolas Papadakis pour les références bibliographiques et les connaissances riches qu'ils ont aimablement partagés avec moi. Je tiens également à remercier Charles Dossal pour l'énergie qu'il sait communiquer mais également l'entrain qu'il a su provoquer autour du travail sur les preuves théoriques de convergence d'algorithme de minimisation de fonctionnelles non-convexes. En espérant que ce projet s'enrichira dans un avenir proche.

Je remercie de manière générale mes collègues, camarades ainsi que les membres de l'équipe Image de l'IMB avec lesquels mes rapports furent enrichissants. En particulier Ioana Ilea, Jean Bourgeois et Cornel Zachiu, qui ont su donner aux pauses café et aux repas du restaurant universitaire la pointe de saveur qui leurs manque. Pauses café au cours desquelles j'ai pu rencontrer d'anciens professeurs, discuter de manière informelle

avec les membres de l'équipe image, aller à la rencontre des jeunes stagiaires de M1 et L3, bavarder avec les doctorants... Je salue également l'efficacité admirable et souriante du secrétariat et de l'équipe informatique.

En particulier, je félicite Pierrick Quémar pour la qualité de son stage de Licence. Je lui souhaite réussite dans la suite de ses études.

Je tiens à m'excuser auprès de Philippe Jaming de m'être désisté vis-à-vis de l'offre de stage et de thèse qu'il m'avait faite. Je veux tout de même le remercier de cette proposition aimable et scientifiquement intéressante.

Je termine en saluant Jean-François Giovannelli pour les conseils prodigués pour la rédaction du rapport de stage. En espérant que ceux-ci auront été respectés au mieux dans les pages qui suivront.

Si l'on devait résumer mes sentiments et mon impression de ce stage en une phrase, ce serait par la citation du philosophe :

> Ce n'est qu'au crépuscule que la chouette de Minerve prend son envol.
>
> « Principes de la philosophie et du droit. »Hegel.

Cette citation reflète ce que j'ai appris avec étonnement durant ces mois de stage dans un laboratoire de recherche. En dépit du fait qu'un rapport de stage, qu'un mémoire de recherche, ou plus généralement qu'un article scientifique soit (dans la plupart des cas) un document formel et bien organisé, doté d'une structure académique, logique ou parfois même pédagogique, le travail et les efforts fournis pour y parvenir, les pistes de réflexion employées sont extrêmement tortueuses.

Enfin j'ai pu constater que la recherche est aussi une affaire de chance puisque c'est par hasard que j'ai découvert une partie des résultats et des techniques présentées dans ce rapport. Cette chance qui a su mettre sur le chemin de ma réflexion les bonnes choses et les bonnes personnes, aux bons moments et aux bons endroits.

Table des matières

1 Introduction. 11

2 Étude bibliographique. 13
2.1 État de l'art en colorisation. . . . 13
2.1.1 Méthodes manuelles. . . . 13
2.1.2 Méthodes basée sur une image exemple (ou basée-exemple). . . . 15
2.2 Les espaces de couleurs usuels et usités. . . . 18
2.2.1 Les espaces linéaires par rapport à RGB. . . . 19
2.2.2 Deux exemples d'espace non linéaire : l'espace Lab et $l\alpha\beta$. . . . 20
2.3 Optimisation convexe. . . . 21
2.3.1 Notion de sous-différentielle. . . . 21
2.3.2 Projection et opérateurs proximaux. . . . 21
2.3.3 L'algorithme de Forward-Backward. . . . 22
2.3.4 Dualité et algorithme primal-dual. . . . 23
2.3.5 Cas de la variation totale. . . . 23
2.3.6 La variation totale couleur spatialement couplée. . . . 25
2.4 La variation totale vectorisée. . . . 25
2.4.1 Motivations. . . . 25
2.4.2 Outils théoriques. . . . 27
2.4.3 Calcul de la projection. . . . 31
2.5 Étude du modèle de Bugeau, Ta et Papadakis [1]. . . . 32

3 Évolution du modèle et contributions. 37
3.1 Évolution du modèle vers l'espace RGB. . . . 37
3.1.1 Inconvénients et avantages de l'espace YUV dans le problème de la colorisation basée-exemple. . . . 37
3.1.2 Étude qualitative et comparative des fonctionnelles. . . . 37
3.2 Résolution par algorithme primal-dual. . . . 38
3.2.1 Positionnement du problème. . . . 38
3.2.2 Le problème de la luminance et des bornes. . . . 38
3.2.3 Mise en place à proprement parler de l'algorithme primal-dual. . . . 41
3.2.4 Étude des opérateurs. . . . 43
3.2.5 Mise en place dans le cas non-convexe. . . . 43
3.3 Résultats numériques. . . . 44
3.3.1 Observation des résultats. . . . 44
3.3.2 Convergence empirique de l'algorithme. . . . 46
3.3.3 Analyse et comparaison avec les résultats de Bugeau *et al.* [1]. . . . 47
3.3.4 Divers exemples d'images colorisées. . . . 49
3.4 Proposition de critères numériques de qualité de colorisation et comparaison. 54
3.5 Quelques améliorations. . . . 57
3.5.1 Problèmes liés à la non-convexité. . . . 57
3.5.2 Le problème de l'initialisation. . . . 58
3.5.3 Pré-traitement des candidats. . . . 59
3.6 Critiques des méthodes et justification d'un paradoxe du modèle. . . . 60

4 Conclusion. 62

A Méthodes alternatives testées. 65
A.1 Résolution par Forward-Backward généralisé. . . . 65
A.2 primal-dual par projection approchée 66
A.3 Un deuxième essais de Generalized Forward-Backward (GFB). . . . 68

1 Introduction.

Nous présenterons dans ce rapport un nouveau modèle de colorisation basé-exemple. C'est-à-dire une méthode prenant en entrée une image en couleur et une image en niveaux de gris et retournant l'image en niveaux de gris colorisée. On peut voir cette opération comme un problème inverse mal posé : cela correspond à l'inversion de la fonction Matlab `rgb2gray`, qui transforme une image RGB en une image en niveaux de gris correspondant au canal de luminance $Y = 0.2990R + 0.5870G + 0.1140B$, avec pour *a priori* un vecteur de l'espace initial.

Remarquons que la difficulté essentielle du problème est très facile à vulgariser : supposons que l'on photographie deux crayons de couleurs de même luminance, il sera strictement impossible de les discerner sur l'image en niveaux de gris tandis qu'ils sont différents sur l'image couleur.

Il faut donc tirer un maximum d'informations de cette image en couleur qui en sert de source (on appellera dans tout ce qui suit cette image, image source.) Nous effectuerons cela par une méthode de comparaison de texture. Nous comparerons localement la luminance de l'image source avec ce que nous appellerons l'image cible : l'image en niveaux de gris à coloriser. Le mot localement est à comprendre au sens où nous comparerons des ensembles carrés de pixels, centrés en le pixel d'intérêt. Ces ensembles seront appelés patchs dans la suite.

Les contributions apportées au modèle, que nous présenterons de manière détaillée en Section 2.5, dans ce rapport sont les suivantes :

- Nous transformons le modèle YUV en RGB et nous avons proposé une résolution de la fonctionnelle par algorithme primal-dual (voir Section 3.2).
- Nous modifions le terme de régularisation dans la fonctionnelle pour obtenir un couplage des canaux en position et en direction (voir Section 2.4).
- Nous mettons en place un pré-traitement des données (voir Section 3.5.3).
- Nous améliorons l'initialisation de l'algorithme (voir Section 3.5.2).
- Nous posons la définition d'un critère numérique de qualité (voir Section 3.4).

Les contributions apportées dans ce travail vont faire l'objet d'une publication dans un article de journal international.

Dans un premier temps, nous présenterons brièvement l'état de l'art en colorisation (voir Section 2.1). Ensuite, après quelques rappels sur les espaces couleurs (Section 2.2), l'état de l'art en optimisation convexe (Section 2.3). La fonctionnelle que nous devrons minimiser ne sera pas convexe, mais nous nous servirons tout de même d'algorithmes destinés originellement aux fonctionnelles convexes. Nous définirons ensuite une variation totale spécialement destinée aux images 2-D à trois canaux et détaillerons ses propriétés (Section 2.4). Nous ne détaillerons pas toutes les preuves, certaines n'étant que des applications de techniques d'algèbre standards. Néanmoins nous présenterons les outils qui permettent de comprendre l'implémentation que nous avons utilisée. Nous terminerons l'état de l'art en décrivant de manière exhaustive le modèle (Bugeau *et al.* [1], voir Section 2.5) dont nous nous inspirerons par la suite en émettant quelques premières critiques et analyses. Cette méthode basée-exemple se place parmi les plus efficaces, c'est pourquoi nous nous comparerons dans la suite à cette méthode.

Ensuite nous mettrons en place notre modèle et nous donnerons les raisons qui nous ont naturellement poussés à mettre ce modèle en place et les avantages que nous pourrons en tirer. Puis nous décrirons précisément comment mettre en place l'algorithme à proprement parler, et notamment la façon dont le problème de colorisation se transforme en un élégant problème de géométrie affine de base (Section 3.2.2). Nous mettrons en place l'algorithme primal-dual puis nous analyserons les images obtenues. En particulier nous étudierons la distribution des couleurs après convergence (voir Section 3.4) de l'algorithme de Bugeau *et al.* [1] et le modèle proposé. Nous essayerons divers types d'images (voir Section 3.3). Nous proposerons aussi quelques améliorations au niveau de l'initialisation ainsi qu'un système de pré-traitement (voir Section 3.5.3) et nous modifierons un peu l'initialisation (voir Section 3.5.2). Nous penserons aussi à émettre quelques critiques sur le modèle en lui-même (voir Section 3.6).

Une dernière section (voir Section A) présentera les méthodes que nous avons testées

pour éviter la difficulté que posaient la minimisation de la fonctionnelle, celle-ci comportant trop de termes.

2 Étude bibliographique.

2.1 État de l'art en colorisation.

La colorisation d'image est un sujet relativement ancien. Sa nécessité apparaît naturellement quand le support matériel a permis d'avoir une image en couleur et que les données enregistrées (films) étaient en niveaux de gris car le matériel ne permettait pas d'enregistrer en couleur.

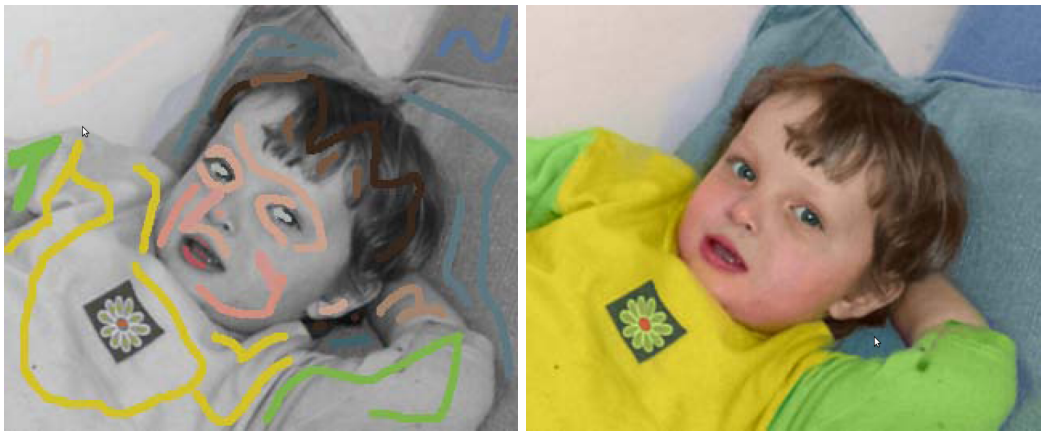
Le problème de colorisation se place donc dans une optique de restauration de documents anciens. Notons également que la colorisation d'image peut aussi être utilisée dans des cas plus basiques d'affichage de données, afin d'en améliorer la compréhension et l'interprétation, par exemple pour différencier plusieurs zones sur des images d'IRM, ou encore dans les aéroports, sur les écrans des détecteurs à rayon X, pour afficher en orange des niveaux de gris correspondant à des densités d'explosifs connus.

Le cas qui nous intéressera pour l'instant est le problème de coloriser une image en niveaux de gris, de manière à respecter une cohérence visuelle, et de conserver des couleurs qui paraissent naturelles pour l'œil humain. Pour cela divers algorithmes existent déjà. Tous requièrent une intervention humaine. Ce qui est parfaitement normal, puisque la solution que nous recherchons se trouvent sur une partie d'un $\mathbb{R}-$espace vectoriel de grande dimension, et admet par cela un nombre immense de solutions. Pour pallier ce problème, on ajoute une information que l'on peut qualifier d'*a priori*. Il s'agit de poser un modèle permettant d'établir un lien entre une information sur le niveau de gris et des points de l'espace des couleurs. On peut ajouter à cela un dispositif permettant d'avoir une cohérence spatiale des couleurs.

On peut également imaginer ce problème comme un problème inverse, où l'opérateur à inverser est concrétisé par la fonction Matlab `rgb2gray` qui transforme une image couleur en niveaux de gris.

2.1.1 Méthodes manuelles.

Ces méthodes requièrent une intervention de l'utilisateur. Celui-ci doit poser des points de couleurs (appelés parfois scribbles) sur l'image en niveaux de gris (voir Figure 2). Le travail effectué par l'algorithme est un travail de cohérence spatiale (et temporelle dans le cas d'une colorisation de vidéo).



(a) Points posés sur l'image.

(b) Image colorisée par la méthode de Levin *et al.* [2].

FIGURE 2 – Exemple de méthode manuelle.

Les travaux précurseurs dans ce domaine sont les travaux de Wilson Markle en 1970 dont le fonctionnement est protégé par un brevet [3].

Parmi les travaux plus récents, on peut lister quelques algorithmes existant et dont le fonctionnement est bien décrit. Citons pour commencer l'algorithme de Levin *et al.* [2], pour lequel le lien entre les niveaux de gris et la couleur est effectuée par l'utilisateur, manuellement, en déposant à l'aide d'une interface des points de couleurs (des scribbles, ou germes). Puis on laisse un procédé de cohérence spatiale coloriser l'image en émettant l'hypothèse que si le niveau de gris varie peu, la couleur varie peu. Précisément, on va minimiser la fonctionnelle

$$H(U) = \sum_r \left(U(r) - \sum_{r \sim s} w_{rs} U(s) \right)^2 , \tag{1}$$

où $r \sim s$ signifie que les pixels r et s sont voisins, et U un canal de chrominance (on minimise la même fonctionnelle pour l'autre). Les w_{rs} sont des poids, qui peuvent être soit :

$$w_{rs} \propto e^{(Y(r)-Y(s)^2/2\sigma^2} ,$$

soit :

$$w_{rs} \propto 1 + \frac{1}{\sigma_r^2}(Y(r) - \mu_r)(Y(s) - \mu_r) ,$$

où μ_r et σ_r représentent la moyenne et la variance dans un voisinage du pixel r.

Ainsi, l'algorithme propage les couleurs définies par les germes à toute l'image. Les problèmes de cet algorithme sont essentiellement dûs à son principe même : l'utilisateur devant poser des germes, s'il y a beaucoup de couleurs en présence, l'utilisateur devra poser beaucoup de germes de manière à avoir un résultat qui ne soit pas anormalement simpliste quant à la diversité des couleurs. Cela peut donc demander, pour des images complexes, un travail fastidieux dont on veut s'affranchir : pour cela on peut consulter dans la Section 2.1.2, la méthode d'Irony *et al.* [4].

Nie *et al.* [5] propose une amélioration de la méthode de Levin *et al.* [2] qui reste une colorisation manuelle. Il s'agit de diviser récursivement l'image en quatre rectangles puis appliquer cette méthode jusqu'à obtenir des carrés trop petits, soit avoir une dynamique sur le carré qui soit plus petite qu'un certain seuil. Cette division de l'image en carrés est appelée décomposition en quadtree (voir figure 3).

(a) Lena en niveaux de gris.

(b) Lena décomposée en quadtree, donnée par Nie *et al.* [5].

FIGURE 3 – Exemple de décomposition en quadtree.

Comme dans le modèle de Levin *et al.* [2] on minimisera la fonctionnelle $H(u) = \sum_r \left(U(r) - \sum_{r \sim s} w_{rs} U(s)\right)^2$. Néanmoins, la définition de voisin est différente : on dira que

deux points sont voisins si les carrés auxquels ils appartiennent sont dans des quadtrees connectés. On modifie également les poids w_{pq} en posant

$$w_{rs} = \begin{cases} \dfrac{1}{|y(r) - y(s)| + \varepsilon} & \text{si } |y(r) - y(s)| < T_w \\ 0 & \text{sinon.} \end{cases}$$

T_w est un seuil qui dépend de la texture du quadtree pour éviter la diffusion de couleur entre quadtree de textures différentes.

On peut également citer deux articles décrivant des algorithmes travaillant en RGB. Il s'agit de Horiuchi *et al.* [6] et Takama *et al.* [7], le deuxième étant une amélioration du premier. Des points sont posés par l'utilisateur. Ce dernier choisit en certains pixels une teinte qui lui est proposée, ces teintes représentent l'ensemble des couleurs que l'on peut réaliser pour la luminance donnée par le pixel à coloriser. Ces couleurs existent sur un plan affine. On réalise ensuite une diffusion de la manière suivante : pour chaque pixel voisin d'intensité Y d'un pixel déjà colorisé, on cherche parmi tout les vecteurs RGB de luminance égale à Y le vecteur RGB le plus proche du pixel colorisé au sens de la distance euclidienne. La version de Takahama *et al.* [7] est une amélioration : la méthode d'Horiuchi *et al.* ne prend pas en compte les contours où peuvent apparaître des différences de teinte. Takahama *et al.* [7] proposent une segmentation en se plaçant dans l'espace colorimétrique Lab afin de détecter la présence de contours et modifier ou arrêter la diffusion de couleurs lors d'une rencontre avec un contour, ce qui améliore les résultats en limitant les phénomènes de bavure.

Une méthode rapide a été proposée par Yatziv et Sapiro [8]. Il s'agit d'une méthode où l'utilisateur donne les chrominances pour certains pixels. Puis pour chaque pixel de l'image en niveaux de gris on calcule une distance entre le point à coloriser et chacun des points définis par l'utilisateur, prenant en compte les variations de niveau de gris sur chacun des chemins parcourus, puis on donne une moyenne pondérée des diverses chrominances, par une fonction de poids ayant des propriétés asymptotiques analogues à la fonction inverse, pour rappel :

- $\lim_{r \to 0} w(r) = \infty$;
- $\lim_{r \to \infty} w(r) = 0$;
- $\lim_{r \to \infty} w(r + r_0)/W(r) = 1$.

Yatziv *et al.* proposent des résultats expérimentaux avec la fonction $\dfrac{1}{r^b}$ avec $1 \leqslant b \leqslant 6$.

On peut résumer les techniques déjà existantes dans le tableau 1.

2.1.2 Méthodes basée sur une image exemple (ou basée-exemple).

Une première méthode basée sur une image d'exemple est présentée par Welsh *et al.* [10], où il n'y a pas de cohérence spatiale recherchée. On recherche pour chaque pixel de l'image cible, quel pixel de l'image source a la luminance et l'écart-type, dans un voisinage de taille 5×5 pixels, le plus proche. Le pixel est alors colorisé par la couleur trouvée (voir figure 4).

Welsh *et al.* propose également à l'utilisateur de limiter le lieu de recherche pour la couleur sur l'image source. Une amélioration est proposée par Di Blasi *et al.* [11] pour accélérer la recherche de voisinage proche par un algorithme de tree-clustering. L'algorithme de Welsh *et al.* peut être long (1 heure), on réduit le temps de recherche à quelques minutes à qualité équivalente. L'algorithme d'Irony *et al.* [4] reprend la méthode de Levin *et al.*, ceci afin d'assurer la cohérence spatiale [2]. Celui-ci propose d'initialiser l'algorithme de Levin *et al.* en utilisant une image source, qui servira d'exemple pour choisir les couleurs. Cette technique affranchit l'utilisateur d'un travail laborieux, puisqu'il n'a plus qu'à choisir une image source qui contient l'information nécessaire à la colorisation. Ici, il s'agit de segmenter l'image source et de classifier les zones. Pour initialiser l'algorithme il suffit ensuite de déterminer sur la base de critères texturaux, pour chaque pixel, de quelle zone son voisinage se rapproche le plus, et affecte une moyenne pondérée des couleurs de chaque zone segmentée par un critère de ressemblance avec la texture

TABLE 1 – Récapitulatif des méthodes existantes en colorisation manuelle.

Auteurs.	Cohérence spatiale.	Espace couleur employé.
Levin *et al.* [2] 2004	Luminance peu variable = chrominance peu variable : $H(u) = \sum_r (U(r) - \sum_{r \sim s} w_{rs} U(s))^2$	YUV
Sapiro [9]	Minimisation de la fonctionnelle $\min_{C_b} \int_\Omega \rho(\|\|\nabla Y - \nabla B_b\|\|) d\Omega$, avec ρ une norme-2, ou une norme-1.	$YCbCr$
Horiuchi [6]	Diffusion markovienne.	RGB
Takahama *et al.* [7]	Diffusion markovienne et segmentation dans l'espace Lab.	RGB
Yatziv *et al.* [8]	On définit une distance prenant en compte les irrégularités de l'image qu'elle traverse, et on pondère les chrominances de chaque points de départ par une fonction de pondération qui tient compte de ces distances.	$YCbCr$
Nie *et al.* [5]	Amélioration de Levin *et al.* [2], par découpage de l'image en parties homogènes.	YUV

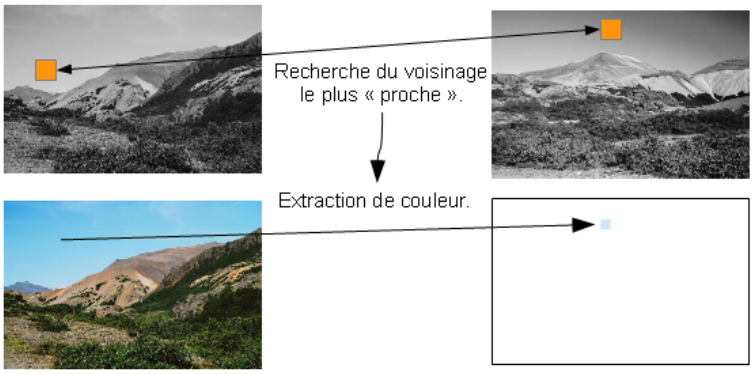


FIGURE 4 – On compare les voisinages du pixel d'intérêt avec ceux des pixels de l'image source, puis on en déduit une couleur pour l'image à coloriser en le pixel d'intérêt.

d'une zone segmentée. Les limites de cette technique sont de trouver un bon algorithme de segmentation, et d'avoir une image source dont les couleurs soient homogènes sur les zones de textures similaires.

L'algorithme de Charpiat *et al.* [12] est en cela plus intéressant car il ne nécessite vraiment qu'une intervention limitée de l'utilisateur, sinon que de fournir une image source. L'algorithme compare alors les 27 critères texturaux les plus représentatifs sur les 64 que permet le modèle SURF [13] pour chaque pixel puis retient les couleurs les plus probables. Intervient alors la cohérence spatiale : on introduit une norme euclidienne de l'espace Lab entre 8-voisins. On commence par résoudre le problème par graph-cut (mincut-maxflow) qui initialise un algorithme de descente de gradient. L'inconvénient majeur de cette méthode est d'être lourd en terme de techniques employées. De plus pour ne pas le complexifier d'avantage, ses auteurs n'ont pas voulu coloriser les bords de

l'image, ce qui est un inconvénient en pratique. Remarquons que l'on pourrait palier ce problème en diminuant les descripteurs, mais le nombre de descripteurs est si grand, qu'il est difficile de savoir si les descripteurs sont utiles ou non aux bords.

Chen *et al.* [14] proposent une méthode bayésienne. La méthode commence par segmentation par soustraction de fond (en anglais 'image matting') à l'aide d'un modèle bayésien. Il s'agit d'un modèle de vraisemblance gaussien tenant compte des variations inter pixel. En suite, l'inférence s'effectue de la même manière que dans la méthode de Welsh *et al.* [10]. Chen fait remarquer dans son article que les visages sont particulièrement difficiles à coloriser à cause des parties lisses entrecoupées de contours [1].

Le modèle de Bugeau, Ta et Papadakis [1] a l'avantage d'être plus simple car il comporte nettement moins de descripteurs : 3 descripteurs sont retenus. Puis la cohérence spatiale limite les erreurs et les artefacts : on régularise l'image couleur en minimisant sa variation totale en U et V (canaux de chrominance), ce qui permet d'éliminer les artefacts en conservant des zones de contour de l'image. On décrira cette méthode en détail dans la section 2.5.

Le tableau 2 récapitule les méthodes baseés-exeemple.

1. Notons ici que notre modèle fonctionnant par minimisation de la variation totale, il sera exempt de ce type de problèmes. Nous testerons notre algorithme sur des portraits pour vérifier.

TABLE 2 – Tableau récapitulatif des méthodes basées-exemple existant dans l'état de l'art.

Auteurs.	Attache aux données entre couleurs et niveaux de gris (inférence.)	Cohérence spatiale.	Espace couleur employé.
Irony *et al.* [4] 2005	On découpe l'image source en zones et on recherche à quelle zone se rapporte le pixel gris à partir d'une analyse harmonique sur les pixels voisins.	idem que Levin. *et al.* [2] (cf tableau 1)	YUV
Welsh *et al.* [10] 2002	Couleur la plus proche suivant un critère de luminance et de texture de patch entourant le pixel (écart type sur un voisinage 5×5) + limitation de la zone de recherche de patch par l'utilisateur.	Aucune, sinon le choix de l'utilisateur.	$l\alpha\beta$
Di Blasi *et al.* [11] 2003	idem que Welsh *et al.*	idem que Welsh *et al.*	YUV
Charpiat *et al.* [12] 2008	Candidats sélectionnés suivant les textures : les 27 attributs les plus importants (déterminés par ACP) parmi les 64 que permet le modèle SURF.	Norme Lab sur les voisins 8 connexes. (initialisé par un résultat d'une optimisation discrète obtenue par graph-cut suivi d'une optimisation continue par descente de gradient.)	Lab
Chen *et al.* [14]	Idem que Welsh *et al.* [10].	Segmentation par soustraction de fond (en anglais 'image matting') à l'aide d'un modèle bayésien.	$l\alpha\beta$
Bugeau, Ta, Papadakis. [1] 2013	Candidats sélectionnés sur l'image source selon leurs caractéristiques de texture.	TV.	YUV

2.2 Les espaces de couleurs usuels et usités.

Remarquons que les diverses techniques de colorisation sont pour la plupart mises en place sur des espaces de couleurs différents de l'espace RGB. On dispose dans nos données initiales d'une image en niveaux de gris dont on a supposé qu'il s'agissait du canal de luminance de notre image à coloriser. On dispose donc, puisqu'une couleur est représentée par trois valeurs, d'une des trois valeurs. On doit alors trouver deux autres valeurs. On peut donc se placer sur un espace couleur dont un des canaux est la luminance. Cela procure un double avantage : d'une part, les algorithmes ne modifient pas le canal de luminance, ce qui permet d'avoir la contrainte de luminosité respectée de manière gratuite. De plus, l'œil humain étant peu sensible aux canaux de chrominance, et d'avantage à la luminance, les éventuelles faiblesses de l'algorithme peuvent passer inaperçues.

2.2.1 Les espaces linéaires par rapport à RGB.

Commençons par citer les espaces dont les coordonnées sont linéaires par rapport à celles de l'espace RGB. L'espace YUV, utilisé dans le standard vidéo couleur PAL (Allemagne, Chine, ...) et SECAM (France, Russie,[2]...), l'espace YIQ, utilisé dans le standard vidéo couleur NTSC (USA, Canada, ...) et l'espace $YCbCr$, utilisé dans le standard de compression JPEG. Ces espaces ont de commun la coordonnée de luminance [3] (Y), définie pour chacun des modèles comme étant égale à $0.2990R + 0.5870G + 0.1140B$. Ces systèmes sont définis de manière à avoir un canal Y auquel l'œil est sensible, et des canaux auxquels l'œil réagit moins bien (chrominance). Dans le cas de la colorisation on considère que l'image cible (à coloriser) représente la luminance, donc pour coloriser, il peut parfois être plus simple de ne considérer que le calcul des canaux de chrominance. Les transformations des coordonnées RGB vers celles de tels espaces sont linéaires et sont définies par des matrices de coefficients empiriques. Pour YUV, espace couleur :
$R, G, B, Y \in [0, 1]$, $U \in [-0.436, 0.436]$, $V \in [-0.615, 0.615]$.

$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0,299 & 0,587 & 0,114 \\ -0,14713 & -0,28886 & 0,436 \\ 0,615 & -0,51498 & -0,10001 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}.$$

En inversant la matrice :

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1,13983 \\ 1 & -0,39465 & -0,58060 \\ 1 & 2,03211 & 0 \end{pmatrix} \begin{pmatrix} Y \\ U \\ V \end{pmatrix}.$$

Pour $YCbCr$:

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.1687 & -0.3313 & 0.5 \\ 0.5 & -0.4187 & -0.0813 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 0 \\ 128 \\ 128 \end{pmatrix}$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.0000 & -0.0000 & 1.4020 \\ 1.0000 & -0.3441 & -0.7141 \\ 1.0000 & 1.7720 & -0.0001 \end{pmatrix} \left(\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} - \begin{pmatrix} 0 \\ 128 \\ 128 \end{pmatrix}. \right)$$

Pour YIQ :

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.595716 & -0.274453 & -0.321263 \\ 0.211456 & -0.522591 & 0.311135 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}.$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1 & 0.9563 & 0.6210 \\ 1 & -0.2721 & -0.6474 \\ 1 & -1.1070 & 1.7046 \end{pmatrix} \begin{pmatrix} Y \\ I \\ Q \end{pmatrix}.$$

L'espace XYZ se veut être un espace tel que la distance perceptuelle entre deux teintes corresponde (approximativement) à une distance euclidienne dans cet espace.

2.

> Retour de Russie, j'expose au Conseil le résultat de mes entretiens sur différents sujets de notre coopération. Je parle du SECAM, de l'accélérateur de particule franco-russe de Sherpoukhov, de notre chambre à bulles Mirabelle. Puis j'élargis le propos : les circonstances sont favorables, les Russes souhaitant développer notre coopération, non seulement dans toutes les recherches appliquées, mais aussi dans le domaine de la production. (en guise de conclusion)« Nous ne ferons rien avec les Russes, comme avec les autres, si nous ne sommes pas capables de faire quelque chose sans eux. »[...] Le Général jubile : « Le moment est venu de vouloir. »
>
> Extrait de *C'était De Gaulle, Volume 3*, Alain Peyrefitte.

3. Attention : Matlab utilise $0.2989R + 0.5870G + 0.1140B$.

On le définit par :

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 2.768892 & 1.751748 & 1.1302 \\ 1.0000 & 4.5907 & 0.0601 \\ 0.000000 & 0.056508 & 5.594292 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}.$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 0.418456 & -0.158657 & -0.082833 \\ -0.091167 & 0.252426 & 0.015707 \\ 0.000921 & -0.002550 & 0.178595 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}.$$

Cet espace linéaire est néanmoins limité par une réponse non linéaire de l'œil, ce qui amène à introduire des espaces non linéaires.

2.2.2 Deux exemples d'espace non linéaire : l'espace Lab et $l\alpha\beta$.

L'espace Lab (plus précisément CIE[4] $L*a*b*$) dérive de l'espace XYZ qui est une transformation linéaire de RGB. Puis on passe de XYZ à Lab par une fonction à l'expression algébrique par morceaux. L'avantage de l'espace Lab est que les couleurs proches dans cet espace pour la norme euclidienne sont des couleurs visuellement proches.

$$\begin{aligned} L^\star &= 116 f(Y/Y_n) - 16, \\ a^\star &= 500\left[f(X/X_n) - f(Y/Y_n)\right], \\ b^\star &= 200\left[f(Y/Y_n) - f(Z/Z_n)\right], \end{aligned}$$

avec

$$f(t) = \begin{cases} t^{1/3} & \text{si } t > (\frac{6}{29})^3, \\ \frac{1}{3}\left(\frac{29}{6}\right)^2 t + \frac{4}{29} & \text{sinon.} \end{cases}.$$

Ici X_n, Y_n et Z_n sont les composantes du blanc de référence (blanc décrit dans l'espace XYZ, n pour neutre).

Un autre espace est important : l'espace $l\alpha\beta$ [15], utilisé par Welsh [10]. Sur cet espace les canaux de couleurs sont décorrélés.

Cet espace est obtenu à partir de l'espace XYZ. On commence par calculer les coordonnées dans l'espace LMS :

$$\begin{pmatrix} L \\ M \\ S \end{pmatrix} = \begin{pmatrix} 0.3811 & 0.5783 & 0.0402 \\ 0.1967 & 0.7244 & 0.0782 \\ 0.0241 & 0.1288 & 0.8444 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}.$$

Puis on calcule $\mathcal{LMS}$:

$$\begin{aligned} \mathcal{L} &= \log(L)\ , \\ \mathcal{M} &= \log(M)\ , \\ \mathcal{S} &= \log(S). \end{aligned}$$

Enfin, on effectue une dernière transformation linéaire :

$$\begin{pmatrix} l \\ \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{3}} & 0 & 0 \\ 0 & \frac{1}{\sqrt{6}} & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & -2 \\ 1 & -1 & 0 \end{pmatrix} \begin{pmatrix} \mathcal{L} \\ \mathcal{M} \\ \mathcal{S} \end{pmatrix}.$$

4. La Commission internationale de l'éclairage (CIE) est une organisation internationale dédiée à la lumière, l'éclairage, la couleur, les espaces de couleur. Elle a été fondée à Berlin en 1913 et est actuellement basée à Vienne en Autriche.

2.3 Optimisation convexe.

Bien que le problème de colorisation ne soit pas convexe par nature, il nous faudra pourtant écrire des algorithmes qui fonctionnent pour minimiser des fonctionnelles. Tout d'abord, notre modèle sera convexe en l'une de ses variables, donc on pourra utiliser des algorithmes d'optimisation convexe. Commençons par citer l'algorithme de Forward-Backward de Combettes et Pesquet [16] qui permet de minimiser des fonctionnelles s'exprimant comme somme de fonctions lisses (différentiables) et d'une fonction semi-continue, convexe propre, *i.e.* de fonctions pour lesquelles il existe un opérateur proximal [17].

2.3.1 Notion de sous-différentielle.

Définition 2.1 (La sous-différentielle.)**.** Dans le cas de fonctions convexes non lisses, on peut définir une notion faible de dérivée.

Soit Γ_0 l'ensemble des fonctions convexes de $\mathbb{R}^N \mapsto \mathbb{R} \cup \{+\infty\}$ semi-continue inférieurement, *i.e.*

$$\Gamma_0 := \{f \text{ tq } \forall x_0 \in \mathbb{R}^N \ , \ \forall \epsilon > 0 \ , \ \exists V \text{ voisinage de } x_0 \ , \ \forall x \in V : f(x) \leqslant f(x_0) - \epsilon\} \ .$$

On définit [18] la sous-différentielle au point x, l'ensemble

$$\partial f(x) := \{u \in \mathbb{R}^N \text{ tq } \forall y \in \mathbb{R}^N \ , \ \langle y - x | u \rangle + f(x) \leqslant f(y)\} \ .$$

Exemple 2.2. *L'application $f : x \mapsto |x|$ est convexe, continue donc semi-continue inférieurement. Et on a : $\partial f(0) = [-1, 1]$, $\partial f(x) = \{1\}$ si $x > 0$ et $\partial f(x) = \{-1\}$ si $x < 0$.*

Théorème 2.3 (Caractérisation d'un minimum.)**.** *Soit J une fonctionnelle convexe semi-continue inférieurement, J admet un minimum en u_0 si et seulement si $0 \in \partial J(u_0)$.*

Démonstration. On trouvera la preuve dans [18]. □

2.3.2 Projection et opérateurs proximaux.

Soit C un ensemble connexe fermé non vide. On définit par

$$1_C(x) := \begin{cases} 0 \text{ si } x \in C \\ +\infty \text{ sinon.} \end{cases}$$

La projection $P_C(x)$ de l'élément x sur le convexe fermé C est la solution u_0 du problème

$$\min_{y \in \mathbb{R}^N} 1_C(y) + \frac{1}{2}||x - y||^2.$$

L'indicatrice 1_C est une fonction de Γ_0.

On étend donc cette notion en définissant l'opérateur proximal d'un fonction semi-continue inférieurement :

Définition 2.4 (Opérateur proximal.)**.** Soient $f \in \Gamma_0$, $x \in \mathbb{R}^N$ le problème suivant a une unique solution :

$$\min_{y \in \mathbb{R}^N} f(y) + \frac{1}{2}||x - y||^2,$$

notée $\text{prox}_f(x)$ appelée l'*opérateur proximal* de f.

Proposition 2.5. *Soit $f \in \Gamma_0$. L'opérateur proximal de f est caractérisé par :*

$$\forall (x, y) \in \mathbb{R}^N \times \mathbb{R}^N \ , \ y = \text{prox}_f(x) \Leftrightarrow x - y \in \partial f(y).$$

REMARQUE : 2.6. Dans ce cas particulier, l'opérateur proximal peut prendre la notation suivante :

$$z := \text{prox}_f(x) = (I_d + \partial f)^{-1}(x).$$

Posons pour cela z tel que

$$f(z) + \frac{1}{2}||x - z||^2 = \min_{y \in \mathbb{R}^N} f(y) + \frac{1}{2}||x - y||^2.$$

Alors on a $z := \text{prox}_f(x)$ et donc $x - z \in \partial f(z)$. Ou encore $x \in z + \partial f(z)$. Ce que l'on note

$$x = (I_d + \partial f(z)).$$

On note cela formellement :

$$z = (I_d + \partial f)^{-1}(x).$$

REMARQUE : 2.7. Dans le cas où f est différentiable il vient :

$$\forall (x, y) \in \mathbb{R}^N \times \mathbb{R}^N \ , \ y = \text{prox}_f(x) \Leftrightarrow x - y = \nabla f(y).$$

2.3.3 L'algorithme de Forward-Backward.

Le cadre d'application de l'algorithme de Forward-Backward revient de manière générale à minimiser le problème d'optimisation convexe de la forme :

$$J(v_0) = \min_{v \in \mathbb{R}^n} f_0(v) + f_1(v).$$

L'algorithme de Forward-Backward va consister à transformer le problème de minimisation de chaque fonctionnelle en projection de la condition initiale sur un sous espace convexe bien choisi.

On a un algorithme, dit de **Forward-Backward**, donné par :

$$x_{n+1} \leftarrow \pi_{\lambda K}(x - \gamma_n \nabla f_1(x_n)).$$

Notons également qu'il existe un algorithme dans le cas où f_0 et f_1 appartiennent à Γ_0 : il s'agit de l'algorithme de Douglas-Rachford (voir Combettes et al [19]).

Théorème 2.8. *Soient $f_0 \in \Gamma_0$ et $f_1 : \mathbb{R}^N \to \mathbb{R}$ convexe et différentiable telle que :*

$$\forall x, y \in \mathbb{R}^N \ , \ ||\nabla f_1(x) - \nabla f_1(y)|| \leqslant \beta ||x - y|| \ ,$$

où $\beta > 0$. On suppose aussi que $f_0 + f_1$ est coercive. Alors il est montré dans [16] que le problème

$$J(v_0) = \min_{v \in \mathbb{R}^n} f_0(v) + f_1(v)$$

admet une unique solution caractérisée par l'équation à point fixe

$$x = \text{prox}_{\lambda f_0}(x - \gamma \nabla f_1(x)).$$

Une version généralisée de cet algorithme est le Generalized Forward-Backward de Raguet *et al.*[20] qui permet de minimiser une fonctionnelle qui est la somme d'une fonction différentiable et d'une somme de fonctions semi-continues inférieurement, convexes et propres dans un espace de Hilbert $\mathcal{H}$. On se servira de cet algorithme. Notre fonctionnelle comportera un terme de régularisation par minimisation de la variation totale, donc on aura besoin d'un algorithme efficace permettant de calculer cet opérateur proximal. Pour cela, il y a l'algorithme de point fixe de Chambolle [21], que nous utiliserons, et l'algorithme primal-dual de Chambolle-Pock [22].

On veut minimiser

$$\min_{u \in \mathcal{H}} F(u) + \sum_{i=1}^{r} G_i(u) \ , \tag{2}$$

avec F à gradient $1/\beta-$lipschitzien, et $G_i \in \Gamma_0$.

Sous certaines hypothèses (intervalle de validité I_λ pour λ_n et uniforme convexité de F), l'algorithme converge vers l'unique minimiseur de la fonctionnelle (2).

Algorithme 1 Forward-Backward généralisé de Raguet *et al.*[20] permettant de minimiser 2.

$0 \leqslant p_1, \ldots, p_r \leqslant 1$ tel que $\sum_i p_i = 1$.
$\gamma_n \in]0, 2\beta[$
$\lambda_n \in I_\lambda$
$z_1, \ldots, z_r \in \mathcal{H}^r$
$X \leftarrow \sum_i p_i z_i$
for n$\geqslant$0 **do**
    for i=1 :r **do**
        $$z_i \leftarrow z_i + \lambda_n \left(\operatorname{prox}_{\frac{\gamma_n}{p_i} G_i} (2 - z_i - \gamma_n \nabla F(X)) - X \right)$$
    end for
    $X \leftarrow \sum_i p_i z_i$
end for

2.3.4 Dualité et algorithme primal-dual.

La mise en forme de la fonctionnelle sous forme duale de notre modèle est grandement inspirée de Bugeau, Ta et Papadakis [1]. On se servira donc de l'algorithme primal-dual de Chambolle-Pock [22] pour résoudre notre problème sous la forme duale.

Soit le problème dit primal :

$$\min_{u \in X} \left[F(Kx) + G(x) \right] \ ,$$

où $G : X \to [0, +\infty]$; $F^* : Y \to [0, +\infty]$ est propre, semi-continue inférieurement, et convexe ; K est un opérateur linéaire continu.

On pose le problème dual associé [22] :

$$\min_{u \in X} \max_{p \in Y} \left[\langle Ku | p \rangle_X + G(u) - F^*(y) \right]. \tag{3}$$

Un algorithme de résolution, est donné dans [22] (Algorithme 2).

Algorithme 2 Algorithme primal-dual de Chambolle et Pock [22]

On initialise τ, $\sigma > 0$, $\theta \in [0, 1]$, $(x^0, y^0) \in X \times Y$ et $\overline{x}^0 = x^0$.
for $n \geqslant 0$ **do**
    $y^{n+1} = (I + \sigma \partial F^*)^{-1}(y^n + \sigma K \overline{x}^n)$
    $x^{n+1} = (I + \tau \partial G)^{-1}(x^n - \tau K^* y^{n+1})$
    $\overline{x}^n = x^{n+1} + \theta(x^{n+1} - x^n)$.
end for

2.3.5 Cas de la variation totale.

L'intérêt de la régularisation d'une image par la variation totale est de permettre une bonne préservation des contours [23].

Dans le cas de fonctions continues localement intégrables, la variation totale est définie [24] comme étant égale à

$$J(u) = \sup_{\varphi \in \mathcal{C}^1_C(\Omega) \ , \ \|\varphi\|_\infty \leqslant 1} \int_\Omega u \operatorname{div}(\varphi). \tag{4}$$

En particulier, les fonctions $\mathcal{C}^1$, la variation totale est égale à

$$J(u) = \int_\Omega \sqrt{\frac{\partial u}{\partial x}^2 + \frac{\partial u}{\partial y}^2} \ , \tag{5}$$

ce qui conduit aux discrétisations ci-dessous.

Dans notre cas, on veut minimiser une fonctionnelle de la forme

$$F(u) + J(u) \ , \tag{6}$$

avec :

$$J(u) = \sum_{1 \leqslant i,j \leqslant N} \sqrt{(\nabla u^1_{i,j})^2 + (\nabla u^2_{i,j})^2} \tag{7}$$

Commençons par définir les opérateurs cités :

Définition 2.9. Posons

$$\nabla u_{i,j} = \begin{pmatrix} \begin{cases} u_{i+1,j} - u_{i,j} \text{ si } i < N, \\ 0 \text{ si } i = N, \end{cases} \\ \begin{cases} u_{i,j+1} - u_{i,j} \text{ si } j < N, \\ 0 \text{ si } j = N. \end{cases} \end{pmatrix}.$$

On pose également :

$$div(p)_{i,j} = \begin{cases} p^1_{i,j} - p^1_{i-1,j} \text{ si } 1 < i < N, \\ p^1_{i,j} \text{ si } i = 1, \\ -p^1_{i-1,j} \text{ si } i = N, \end{cases} + \begin{cases} p^2_{i,j} - p^2_{i,j-1} \text{ si } 1 < j < N, \\ p^2_{i,j} \text{ si } j = 1, \\ -p^2_{i,j-1} \text{ si } j = N, \end{cases} .$$

Lemme 2.10 (Dualité gradient-divergence.)**.**

$$\langle - \operatorname{div} p | u \rangle = \langle p | \nabla u \rangle \, .$$

La démonstration est uniquement calculatoire et tombe juste car la divergence a été définie pour cela.

Cette formule est l'équivalent discret d'une formule découlant de la formule de Stokes appelée parfois formule du gradient. Soit $f \in C^1$, $\overrightarrow{A}$ un champ de vecteur :

$$\int_{\mathbb{R}^n} f \operatorname{div} \overrightarrow{A} \mathrm{d}\lambda_n(x) = - \int_{\mathbb{R}^n} \nabla f \cdot \overrightarrow{A} \mathrm{d}\lambda_n(x)$$

On veut minimiser

$$J(u) = \sum_{1 \leqslant i,j \leqslant N} \sqrt{\left(\nabla u^{(1)}_{i,j}\right)^2 + \left(\nabla u^{(2)}_{i,j}\right)^2}. \tag{8}$$

Proposition 2.11.

$$\begin{aligned} J(u) &:= \sum_{1 \leqslant i,j \leqslant N} \sqrt{\left(\nabla u^{(1)}_{i,j}\right)^2 + \left(\nabla u^{(2)}_{i,j}\right)^2} \\ &= \sup_{\{p \ tq \ |p_{i,j}| \leqslant 1\}} \langle p | \nabla u \rangle \, . \end{aligned}$$

Démonstration. En effet, on a par Cauchy-Schwarz :

$$p^{(1)}_{i,j} \nabla u^{(1)}_{i,j} + p^{(2)}_{i,j} \nabla u^{(2)}_{i,j} \leqslant |\nabla u_{i,j}||p_{i,j}|,$$

et si $|p_{i,j}| = 1$, on a

$$p^{(1)}_{i,j} \nabla u^{(1)}_{i,j} + p^{(2)}_{i,j} \nabla u^{(2)}_{i,j} \leqslant |\nabla u_{i,j}|,$$

d'où

$$\sum_{i,j} p^{(1)}_{i,j} \nabla u^{(1)}_{i,j} + p^{(2)}_{i,j} \nabla u^{(2)}_{i,j} \leqslant \sum_{i,j} |\nabla u_{i,j}|$$

et

$$\sup_{\{p \text{ tq } |p_{i,j}| \leqslant 1\}} \sum_{i,j} p^{(1)}_{i,j} \nabla u^{(1)}_{i,j} + p^{(2)}_{i,j} \nabla u^{(2)}_{i,j} \leqslant \sum_{i,j} |\nabla u_{i,j}|.$$

Et on obtient le cas d'égalité, si $|\nabla u_{i,j}| \neq 0$ grâce à $p_{i,j} = \dfrac{\nabla u_{i,j}}{|\nabla u_{i,j}|}$. Si $|\nabla u_{i,j}| = 0$, l'égalité est immédiatement vérifiée. □

Grâce au Lemme 2.10 une dualité au sens de l'analyse convexe (voir [18] Définition 1.1.1).

Corollaire 2.12. *On déduit du Lemme 2.10 que*

$$J(u) = \sup_{\{\operatorname{div} p \ tq \ |p_{i,j}|\leqslant 1 \ , \ \forall i,j=1,\cdots,N\}} \langle p|u\rangle \ .$$

Ainsi l'équation 6 devient :

$$\min_{u\in X} \max_{p\in Y} \left[- \langle u| \operatorname{div} p\rangle_X + F(U) - \delta_P(p)\right] \ , \tag{9}$$

avec

$$\delta_P(p) = \begin{cases} 0 \text{ si } p \in P \ , \\ +\infty \text{ sinon,} \end{cases}$$

et $P = \{p \in Y : ||p||_2 \leqslant 1\}$.

2.3.6 La variation totale couleur spatialement couplée.

Dans le cas des images en couleur *RGB* on définit une variation totale des images couleur, spatialement couplée :

$$TV(u) = \int_\Omega \sqrt{\frac{\partial u_R}{\partial x}^2 + \frac{\partial u_R}{\partial y}^2 + \frac{\partial u_G}{\partial x}^2 + \frac{\partial u_G}{\partial y}^2 + \frac{\partial u_B}{\partial x}^2 + \frac{\partial u_B}{\partial y}^2 +} \tag{10}$$

Définie de cette manière, les canaux sont couplés dans le sens où les sauts de discontinuité apparaissent aux mêmes endroits sur les trois canaux.

2.4 La variation totale vectorisée.

2.4.1 Motivations.

Pour utiliser une régularisation par variation totale dans le cas d'une image couleur, il existe une variation totale particulière, introduite par Goldluecke *et al.* [25]. On va résumer dans cette partie, le cadre mathématique de cette variation totale en commençant par quelques rappels sur les opérateurs linéaires continus, que l'on utilisera dans un cas très particulier, puis on détaillera un peu l'analyse convexe qui tourne autour de cette variation totale vectorisée.

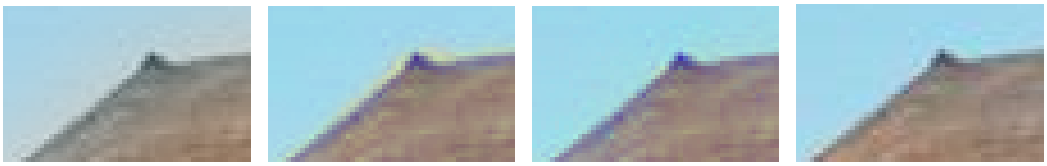
On peut établir l'utilité de cette section « variation totale vectorisée »par des arguments purement géométriques (plutôt « géométrie différentielle »), ce que nous ferons après avoir présenté les avantages numériques que présentent en pratique ces différentes approches. Sur la figure 5, différentes variations totales sont comparées

Pour s'affranchir des erreurs causées par la recherche des candidats par le procédé suivant : on va coloriser une image en niveaux de gris en se servant pour image de référence (image source) la même image. On appellera dans la suite, ce système de test **auto-colorisation**.

On remarque que dans le cas du modèle de Bugeau *et al.* [1], l'image est plus terne, mais ce n'est pas ce qui nous importe le plus pour le moment, nous reviendrons sur ce point en section 2.5. Ce que l'on remarque maintenant, c'est l'effet de halo visible au dessus de la montagne : on remarquera dans la section 2.5, que le modèle de Bugeau *et al.* travaille sur les canaux U et V de chrominance et laisse le canal de luminance constant. Ainsi, les contours au niveau des canaux de luminance et chrominance ne sont pas liés, tant directionnellement que spatialement. Ainsi les contours en chrominance se déplaçant indépendamment de la luminance, il apparait des phénomènes de halo. On peut comparer à la variation totale couplée en position, et on remarque que le phénomène de halo disparait. Il reste néanmoins des problèmes dans un voisinage (réduit) des contours : il apparait des phénomènes de perturbation de l'ordre de 1 à 2 pixels maximum. Ce que fait disparaitre la variation totale dont les directions sont liées. Et cela donne une image plus nette, avec moins de bavures au niveau des contours.

(a) Image originale. (b) Image en niveaux de gris à coloriser (cible).

(c) Modèle Bugeau *et al.* [1]. (d) TV couplée en position. (e) TV couplée en position et direction [25].



(f) Zoom sur 5(c). (g) Zoom sur 5(d). (h) Zoom sur 5(e). (i) Image originale.

FIGURE 5 – Comparaison de différentes régularisations dans le cas de l'auto-colorisation.

Maintenant, abordons la chose avec une vision plus géométrique. Tout d'abord rappelons que les fonctions à variations bornées (*i.e.* les fonctions dont la variation totale est finie), ne sont pas continues, elles peuvent admettre des irrégularités. Ce qui est utile pour faire du traitement d'image. Les images contenant des contours, on peut alors modéliser les images naturelles via les fonctions à variations bornées. Une question naturelle vient alors : comment sont modélisés les contours dans le cas où l'on utilise la variation totale ? En effet, il n'y a pas de fonction de type fonction binaire qui indique « ceci est un contour ». Ces contours sont définis via le gradient : là où le gradient est fort, il y a un contour. Le gradient nous donne de plus une direction de contour (voir figure 6). En effet, soit $\frac{\partial u}{\partial x}(x_0, y_0)$ la dérivée partielle de l'image u au point (x_0, y_0) dans la direction x, (resp. $\frac{\partial u}{\partial y}(x_0, y_0)$, resp. y) et soit e_1 et e_2 la base canonique du plan de l'image, et supposons que Δ soit la tangente de $\mathcal{C}$, le contour de l'objet $\mathcal{O}$, tel que $u = \mathbf{1}_{\mathcal{O}}(x, y)$. Alors Δ est la droite qui passe par (x_0, y_0) et de vecteur normal $\frac{\partial u}{\partial x}(x_0, y_0)e_1 + \frac{\partial u}{\partial y}(x_0, y_0)e_2$.

Définis de cette manière, les contours peuvent donc avoir, en un pixel donné, une direction différente suivant le canal de couleur considéré, même si les canaux sont couplés. Ce couplage ne permettra seulement que des positions (spatiales) communes des contours pour les trois canaux, mais pas nécessairement de même direction. Dans la suite, on définira un procédé permettant d'avoir une direction commune sur chacun des canaux de couleur. On prendra pour cela la direction la plus représentée dans le sens où ce sera la direction dans laquelle la matrice jacobienne aura la plus grande norme d'opérateur (ce que l'on peut interpréter en terme de plus grande dynamique). Cela permettra d'avoir une direction commune pour les trois canaux.

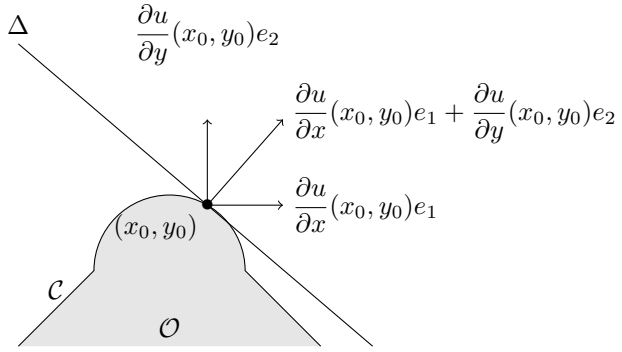


FIGURE 6 – Un contour est défini localement par les dérivées partielles par rapport à chacune des directions.

2.4.2 Outils théoriques.

Cette section suivra la présentation faite dans [25], et visera à expliquer la projection proposée en Algorithme 3.

D'abord, introduisons la notion d'opérateur borné ([26], 4.5.2).

Définition 2.13. On dit qu'un opérateur linéaire d'un espace vectoriel normé E_1 vers E_2 muni chacun d'une norme notées respectivement $||.||_{E_1}$, $||.||_{E_2}$ est borné, s'il existe une constante C telle que :

$$\forall f \in E_1 \ , \ ||Af||_{E_2} \leqslant C||f||_{E_1}. \tag{11}$$

Le plus petit de ces nombres C vérifiant cette inégalité s'appelle norme de l'opérateur A et se note $|||A|||$.

Un théorème caractérise cette norme de manière utile :

Théorème 2.14. *Pour tout opérateur A d'un espace normé dans un espace normé, on a :*

$$|||A||| = \sup_{||x|| \leqslant 1} ||Ax|| = \sup_{||x|| \neq 0} \frac{||Ax||}{||x||}. \tag{12}$$

Démonstration. Voir [26], Section 4.5.2 Théorème 1. □

Un deuxième théorème pourra être utile pour la suite :

Théorème 2.15. *Soit A^t l'opérateur adjoint de A, i.e. tel que*

$$\forall f, g \in (E_1, E_1^*) \ , \ \langle Af | g \rangle = \langle f | A^t g \rangle \ , \tag{13}$$

alors

$$|||A||| = |||A^t|||. \tag{14}$$

Démonstration. Voir [26], Section 4.5.5 Théorème 6. □

Maintenant que l'on dispose des outils suffisants, définissons la variation totale vectorisée [25]. Soit u une application différentiable de $\mathbb{R}^2$ dans $\mathbb{R}^3$ au point (x_0, y_0)

$$\begin{array}{rccl} u: & \mathbb{R}^2 & \to & \mathbb{R}^3 \\ & (x,y) & \mapsto & (u_1(x,y), u_2(x,y), u_3(x,y)) \end{array} \tag{15}$$

On définit la jacobienne de u en (x_0, y_0) par l'application linéaire de $\mathbb{R}^2$ dans $\mathbb{R}^3$ ayant pour matrice dans la base canonique :

$$J(u)_{(x_0,y_0)} = \begin{pmatrix} \frac{\partial u_1}{\partial x}(x_0,y_0) & \frac{\partial u_1}{\partial y}(x_0,y_0) \\ \frac{\partial u_2}{\partial x}(x_0,y_0) & \frac{\partial u_2}{\partial y}(x_0,y_0) \\ \frac{\partial u_3}{\partial x}(x_0,y_0) & \frac{\partial u_3}{\partial y}(x_0,y_0) \end{pmatrix} \tag{16}$$

Dans la suite, on supposera que u est différentiable en tous points, on omettra de préciser (x_0, y_0) et on considérera toujours que la dérivée est prise au point (pixel) courant qui nous intéresse. Cette matrice est la matrice d'un opérateur linéaire borné, et on note sa norme $|||J(u)|||$.

Maintenant que l'on a défini la matrice jacobienne, on peut définir la variation totale vectorisée :

Définition 2.16 (Variation totale vectorisée.)**.** Soit $u \in \mathcal{C}^1(\Omega, \mathbb{R}^{3,2})$. On définit la variation totale vectorisée [25] d'une image u de domaine Ω, et l'on note :

$$TV_J(u) = \int_\Omega |||J(u)_{(\omega_1,\omega_2)}||| \, d\omega. \tag{17}$$

On étendra cette définition plus bas aux fonctions localement intégrables.

Pour calculer effectivement la norme, la proposition suivante peut-être utilisée :

Proposition 2.17. *Soit σ_1 la valeur singulière la plus grande de $J(u)$, i.e. la racine carrée de la plus grande valeur propre de l'application $J(u)^t J(u)$.*

Alors, $|||J(u)||| = \sigma_1$.

Démonstration. Considérons $|||J(u)|||^2$. On a :

$$|||J(u)|||^2 = \sup_{\zeta \text{ tel que } ||\zeta||=1} ||J(u).\zeta||^2. \tag{18}$$

On écrit

$$||J(u).\zeta||^2 = \zeta^t J(u)^t J(u) \zeta. \tag{19}$$

$J(u)^t J(u)$ étant réelle et symétrique, elle se diagonalise dans une base orthonormée ([27], corolaire 8.14), donc il existe P, matrice orthogonale telle que :

$$\zeta^t J(u)^t J(u) \zeta = \zeta^t P^t D P \zeta \ , \tag{20}$$

avec D matrice diagonale contenant les carrés des valeurs singulières, classés par ordre décroissant : σ_i^2.

P étant isométrique pour la norme-2 on obtient donc :

$$\begin{array}{ll} |||J(u)|||^2 & = \sup_{\zeta \text{ tel que } ||\zeta||=1} \zeta^t P^t D P \zeta \\ & = \sup_{\zeta' \text{ tel que } ||\zeta'||=1} {\zeta'}^t D \zeta' \end{array} \ , \tag{21}$$

et le sup est atteint pour $\zeta' = (1, 0, 0)$, et égal à σ_1^2. □

On étend la définition de la variation totale vectorisée en passant à une formulation duale. On définit pour $\xi \in \mathbb{R}^2$, $\mathrm{div}(\xi) = \frac{\partial \xi_1}{\partial x_1} + \frac{\partial \xi_2}{\partial x_2}$.

Proposition 2.18. *Soit $u \in \mathcal{C}^1(\Omega, \mathbb{R}^3)$, on peut écrire :*

$$TV_J(u) = \sup_{\left\{(\xi,\eta) \in \mathcal{C}_C^1(\Omega, E^2 \times E^3)\right\}} \left\{ \sum_{i=1}^{3} \int_\Omega u_i \, \mathrm{div}(\eta_i \xi) \, dx \right\} \ , \tag{22}$$

où E est la boule unité fermée pour la norme-2.

Démonstration. Par définition :

$$|||J(u)||| = \sup_{\xi \in E^2} ||J(u).\xi|| \ , \tag{23}$$

avec E^2 la boule unité fermée pour la norme-2.
Notons que par la formule de Cauchy-Schwarz :

$$\sup_{\eta \in E^3} \langle \eta | J(u).\xi \rangle = ||J(u).\xi||. \tag{24}$$

Donc :

$$|||J(u)||| = \sup_{(\xi,\eta) \in E^2 \times E^3} \sum_{i=1}^{3} \eta_i \sum_{j=1}^{2} \xi_j \partial_j u_i. \tag{25}$$

Avec la dualité gradient-divergence, l'expression devient, en notant . le produit scalaire euclidien :

$$|||J(u)||| = \sup_{(\xi,\eta) \in E^2 \times E^3} -\sum_{i=1}^{3} \eta_i \operatorname{div}(\xi).u_i. \tag{26}$$

Puis, par linéarité de la divergence :

$$|||J(u)||| = \sup_{(\xi,\eta) \in E^2 \times E^3} -\sum_{i=1}^{3} \operatorname{div}(\eta_i \xi).u_i. \tag{27}$$

Par symétrie de la boule unité, et en intégrant sur Ω :

$$\int_\Omega |||J(u)||| = \sup_{(\xi,\eta) \in E^2 \times E^3} \int_\Omega \sum_{i=1}^{3} \operatorname{div}(\eta_i \xi).u_i. \tag{28}$$

□

On peut ainsi avoir une définition duale, comme dans le cas de variation totale en niveaux de gris.

Corollaire 2.19 (Extension de la définition)**.** *Si $u \in L^1_{loc}(\Omega, \mathbb{R}^3)$, l'espace des fonction localement intégrable, on définit la variation totale vectorisée par l'équation* (22)*.*

On va maintenant utiliser des outils de l'algèbre tensorielle afin d'appliquer des résultats bien établis de l'analyse convexe.

Notons que l'on peut transformer deux vecteurs en une matrice de la manière suivante : soit $\eta \in \mathbb{R}^3$ et $\xi \in \mathbb{R}^2$, on note :

$$\eta \otimes \xi = \begin{pmatrix} \eta_1 \xi^t \\ \eta_2 \xi^t \\ \eta_3 \xi^t \end{pmatrix}. \tag{29}$$

On introduit une définition de la divergence sur ces nouveaux objets.

Définition 2.20 (Divergence vectorisée.)**.** Soit $\xi \in \mathbb{R}^2$, on définit :

$$\operatorname{div}(\eta \otimes \xi) := \begin{pmatrix} \operatorname{div}(\eta_1 \xi) \\ \operatorname{div}(\eta_2 \xi) \\ \operatorname{div}(\eta_3 \xi) \end{pmatrix}. \tag{30}$$

De cette manière,

$$\int_\Omega \sum_{i=1}^{3} \operatorname{div}(\eta_i \xi).u_i = \langle u, \operatorname{div}(\eta \otimes \xi) \rangle \, . \tag{31}$$

On a ainsi une autre expression de la variation totale, grâce à ces notations :

Proposition 2.21. *Soit $u \in L^1(\Omega, \mathbb{R}^3)$, on peut écrire :*

$$TV_J(u) = \sup_{v \in \left\{ \operatorname{div}(\eta \otimes \xi) \in \mathcal{C}^1_C(\Omega, conv(E^2 \otimes E^3)) \right\}} \langle u, v \rangle \ , \tag{32}$$

où E est la boule unité fermée pour la norme-1, conv l'enveloppe convexe de $E^2 \otimes E^3$ qui est égale à $E^2 \otimes E^3$, puisque c'est un espace convexe, et $\mathcal{C}^1_C$ l'ensemble des fonctions de classe $\mathcal{C}^1$ à support compact.

Il nous faut définir maintenant une norme qui ait une expression duale.

Définition 2.22 (Norme nucléaire d'une matrice.)**.** Soit A une matrice réelle, on définit la norme nucléaire (notée $|A|_*$) comme la somme de ses valeurs singulières.

Pour définir la dualité, un produit scalaire est nécessaire sur les espaces de matrice.

Définition 2.23 (Produit scalaire de matrices.)**.** Soit $X, Y \in \mathcal{M}_{m,n}$, on définit le produit scalaire entre X et Y :

$$\langle X, Y \rangle = Tr(X'Y) = \sum_{i=1}^{m} \sum_{j=1}^{n} X_{ij} Y_{ij}. \tag{33}$$

Cette dualité a bien un sens pour l'analyse convexe :

Proposition 2.24.

$$|A|_* = \max_{B \in \mathcal{M}_{3,2}} \{ \langle A, B \rangle \ \textit{tel que} \ \sigma_1(B) \leqslant 1 \} \ , \tag{34}$$

où $\sigma_1(B)$ est la plus grande valeur singulière de B.

Démonstration. Voir [28], Proposition 2.1. □

Ce qui permet d'établir le résultat suivant.

Proposition 2.25. *L'enveloppe convexe de $E^2 \otimes E^3$ correspond à la boule unité pour la norme nucléaire :*

$$conv(E^2 \otimes E^3) \{ A \in \mathcal{M}_{3,2} \ \textit{tel que} \ |A|_* \leqslant 1 \} . \tag{35}$$

Introduisons un outil standard de l'analyse convexe :

Définition 2.26 (Fonction de support [18].)**.** Soit S un ensemble non vide de $\mathbb{R}^n$. La fonction $\sigma_S : \mathbb{R}^n \mapsto \mathbb{R} \cup \{+\infty\}$ définie par

$$x \mapsto \sigma_S(x) := \sup_{s \in S} \langle s, x \rangle \ , \tag{36}$$

est appelée fonction de support de S.

On rappelle, avant d'effectuer la preuve de la Proposition 2.25, l'énoncé du théorème de Eidelheit [29], conséquence du théorème de Hahn-Banach :

Théorème 2.27. *Dans un espace normé, tout convexe fermé est l'intersection des demi-espaces fermés qui le contiennent.*

Démonstration de la Proposition 2.25. Commençons par montrer que

$$conv(E^2 \otimes E^3) = \bigcap_{B \in \mathbb{R}^{3\times 2}} \left\{ A \in \mathbb{R}^{3\times 2} \text{ tel que } \langle A, B \rangle \leqslant \sigma_1(B) \right\} \ , \tag{37}$$

où $\sigma_1(B)$ représente la plus grande valeur singulière de B.

Pour cela, remarquons que $E^2 \otimes E^3 \subset \mathbb{R}^{3\times 2}$. Le théorème de Eidelheit nous dit donc que :

$$conv(E^2 \otimes E^3) = \bigcap_{B \in \mathbb{R}^{3\times 2}} \left\{ A \in \mathbb{R}^{3\times 2} \text{ tel que } \langle A, B\rangle \leqslant \sigma_1(B) \right\}. \tag{38}$$

Mais dans ce cas la fonction de support a une expression explicite :

$$\begin{aligned} \sigma_E(B) &= \sup_{\eta\otimes\xi \in E^3\otimes E^2} \langle \eta \otimes \xi, B\rangle \\ &= \sup_{\eta\in E^3, \xi\in E^2} \sum_{i=1}^{2}\sum_{j=1}^{2} \eta_i b_{ij} \xi_j \\ &= \sigma_1(B). \end{aligned} \tag{39}$$

Ce qui montre bien le résultat préliminaire (37).

Ensuite on écrit, via la Proposition 2.24, pour toute matrice $B \in \mathcal{M}_{3,2}$,

$$\langle A, B\rangle \leqslant |A|_*.\sigma_1(B) \ ,$$

où $\sigma_1(B)$ représente la plus grande valeur singulière de B. De plus on a égalité pour un certain B.

Remarquons que $\langle A, B\rangle = |A|_*\sigma_1(B) \Rightarrow \langle A, B\rangle \leqslant |A|_*\sigma_1(B)$, ce qui permet d'écrire que

$$\{A \in \mathcal{M}_{3,2} \langle A, B\rangle = |A|_*\sigma_1(B)\} \subset \{A \in \mathcal{M}_{3,2} \langle A, B\rangle \leqslant |A|_*\sigma_1(B)\}. \tag{40}$$

Ainsi l'intersection

$$conv(E^2 \otimes E^3) = \bigcap_{B \in \mathbb{R}^{3\times 2}} \left\{ A \in \mathbb{R}^{3\times 2} \text{ tel que } \langle A, B\rangle \leqslant \sigma_1(B) \right\} \tag{41}$$

ne se fait que pour les B qui réalisent l'égalité. Ainsi :

$$conv(E^2 \otimes E^3) = \bigcap_{B \in \mathbb{R}^{3\times 2}} \left\{ A \in \mathbb{R}^{3\times 2} \text{ tel que } |A|_*.\sigma_1(B) \leqslant \sigma_1(B) \right\}. \tag{42}$$

On conclut en remarquant que l'ensemble considéré ne dépend plus de B, en simplifiant par $\sigma_1(B)$. □

2.4.3 Calcul de la projection.

On détermine maintenant la projection de la variable duale qui permettra d'effectuer la régularisation.

Théorème 2.28. *Soit $A \in \mathcal{M}_{3,2}(\mathbb{R})$ et soit $A = U\Sigma V$ une décomposition en valeurs singulières, avec Σ et la matrice diagonale ayant (σ_1, σ_2) sur sa diagonale. U et V sont des matrices orthogonales. (Il y a unicité de (σ_1, σ_2) à l'ordre près.)*

Alors, la projection de A sur $conv(E^2 \otimes E^3)$ est donnée par :

$$P(A) = Udiag(\sigma')V \ , \tag{43}$$

avec σ' la projection de (σ_1, σ_2) sur la boule unité pour la norme-1.

Démonstration. Voir [30]. □

Ce théorème permet de donner un algorithme (Algorithme 3) pour calculer la projection 43.

Algorithme 3 Algorithme de projection pour la variation totale vectorisée [25].

1: $A \leftarrow J(u)^t J(u)$
2: Décomposition $A = U \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix} V$
3: $(\sigma_1, \sigma_2) \leftarrow diag(D)$
4: **if** $\sigma_1 + \sigma_2 \geqslant 1$ **then**
5: $\quad (\sigma_1, \sigma_2) \leftarrow \dfrac{(\sigma_1, \sigma_2)}{\sigma_1 + \sigma_2}$
6: **end if**
7: $D \leftarrow \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{pmatrix}$
8: $P(A) \leftarrow UDV$

La variation totale habituelle et la variation totale vectorisée ne sont pas équivalentes mais se comparent :

Proposition 2.29. *Soit $u \in BV(\Omega, \mathbb{R}^2)$,*

$$TV_J(u) \leqslant TV(u). \tag{44}$$

Démonstration. Les termes à intégrer étant positifs, il suffit d'établir l'inégalité en un point. On écrit (l'hypoténuse est le plus grand côté) :

$$\begin{aligned} \sigma_1(J(u)) &\leqslant \sqrt{\sigma_1(J(u))^2 + \sigma_2(J(u))^2} \\ &= \sqrt{|\nabla u_1|_2^2 + |\nabla u_2|_2^2 + |\nabla u_3|_2^2}\ , \end{aligned} \tag{45}$$

car les matrices de la décomposition singulières sont orthogonales.

Et la conclusion vient de la positivité des termes intégrés. □

2.5 Étude du modèle de Bugeau, Ta et Papadakis [1].

Ce modèle servira de point de départ pour la construction de notre modèle.

(a) Image source. (b) Image cible en niveaux de gris.

FIGURE 7 – Données du problème : l'image en couleur, source d'informations, et l'image en niveaux de gris à coloriser.

On suppose que l'image en niveaux de gris représente le canal de luminance définit comme étant égal à $0.299R + 0.587G + 0.114B$ où R, G et B sont les trois primaires rouges, vertes et bleues de l'image cible que l'on recherche. Pour des raisons de commodité on se place sur l'espace YUV pour le problème de colorisation. En effet, une couleur est définie par la donnée de trois valeurs réelles. Afin de conserver une information de niveau de gris, il faut absolument que l'image colorisée (le résultat de l'algorithme) ait un niveau de luminance égal à l'image cible : cela est nécessaire pour obtenir une bonne conservation des textures de l'image colorisée. De manière à faciliter le travail, on peut rechercher

seulement les deux canaux de chrominance de l'image. Cette donnée, cumulée avec celle de la luminance permet de connaitre totalement l'image couleur. Donc l'algorithme se réduit au calcul des canaux de chrominance de l'image cible et non celui des trois primaires.

Sur ce modèle l'attache entre le niveau de gris et la couleur est faite sur la base de critères texturaux. Ils sont plus évolués que ceux utilisés par Welsh *et al.* [10] qui n'utilisent que la variance et le niveau de gris, mais aussi plus simples que ceux utilisés par Charpiat *et al.* [12] (modèle SURF). On retient le meilleur candidat pour :

- l'écart-type sur des tailles de patch, 5×5 et 3×3 ;
- l'amplitude du spectre (FFT) sur des tailles de patch 7×7, 9×9 et 11×11 ;
- L'histogramme cumulé (différence en norme-1) sur des tailles de patch 7×7, 9×9 et 11×11.

Ces critères ont été choisis empiriquement par Bugeau *et al.* [1].

Détaillons ces divers critères. Pour un patch P on pose la distance entre les pixels p et q, :

$$\rho_1(p, q, P) := |\sigma^2(P_p) - \sigma^2(P_q)| \ , \tag{46}$$

où $\sigma^2(P_p)$ (resp. $\sigma^2(P_q)$) représente la variance de la série de donnée que constituent l'ensemble des valeurs de luminance dans le patch P autour du pixel p (resp. q). Pour un pixel p donné, de l'image cible, on recherche dans l'image de luminance de l'image source, les canaux de chrominance du pixel le plus proche au sens de cette distance.

Soit $\hat{P}_p(\xi)$ la transformée de Fourier du patch P autour de p. On pose

$$\rho_2(p, q, P) := \sum_{\xi} \left| ||\hat{P}_p(\xi)||_2 - ||\hat{P}_q(\xi)||_2 \right| . \tag{47}$$

Et de la même manière que précédemment, on recherche le pixel q, le plus proche de p au sens de cette distance.

La dernière distance est équivalente à la distance de Wasserstein : soit $(h_i)_{i=1,\dots,N}$ une partition régulière de $[0, 255]$. On définit l'histogramme H :

$$\begin{array}{rccl} H: & \{1, \dots, N\} & \to & \mathbb{N} \\ & i & \mapsto & \sharp \{x \in \Omega \text{ tel que } f(x) \in h_i\} \ , \end{array} \tag{48}$$

où Ω représente le domaine de l'image. On définit l'histogramme cumulé par

$$\begin{array}{rccl} H: & \{1, \dots, N\} & \to & \mathbb{N} \\ & i & \mapsto & \sum_{j=1}^{i} H(j) \ , \end{array} \tag{49}$$

On pose

$$\rho_3(p, q, P) := \sum_i |H_{P_p}(i) - H_{P_q}(i)|. \tag{50}$$

On suppose maintenant que l'on a une image en niveaux de gris (cible) Ig et une image source de laquelle on a extrait des candidats colorés sur la base de critères texturaux. On dispose donc d'une image en niveaux de gris et pour chacun des pixels de cette image en niveaux de gris un ensemble de 8 couleurs candidates. Il s'agit d'attribuer à chacun des pixels de l'image en niveaux de gris, une couleur. Pour cela, on pose un modèle que l'on peut qualifier de basé-énergie. Il s'agit de minimiser une fonction de coût, ou, par analogie avec les méthodes type moindre-carré, une fonction d'énergie.

Un algorithme de recherche nous donne pour chaque pixel, 8 candidats, que l'on note c_i, i allant de 1 à 8. Une recherche exhaustive serait trop coûteuse et on ne sait pas si le résultat final sera suffisamment lisse, régulier, comme doit l'être une image naturelle. On introduit pour cela une fonctionnelle sur le domaine continu permettant de choisir le candidat tout en cherchant à minimiser la variation totale de l'image. Puisque l'on se place sur un domaine continu, on rend continue l'attache aux données, en introduisant des pondérations w_i devant chaque candidat au niveau de l'attache aux données. On cherche également à éviter les mélanges de deux candidats très différents : pour cela on force les poids w_i à aller vers 0 ou 1.

Le modèle de Bugeau *et al.* est donné par la fonctionnelle suivante, où u est l'image des chrominances que l'on cherche :

$$F(u,W) := TV(u) + \frac{\lambda}{2}\int_\Omega \sum_{i=1}^{8} w_i ||u - c_i||^2 + \frac{\alpha}{2}\int_\Omega \sum_{i=1}^{8} w_i(1-w_i) + \chi_{U\in[0,255]} + \chi_{W\in\Delta}\ , \quad (51)$$

avec

$$TV(u) = \int_\Omega \sqrt{\partial_x U^2 + \partial_y U^2 + \partial_x V^2 + \partial_y V^2}\, d\omega. \quad (52)$$

Ce problème, bien que non-convexe est résolu par un algorithme primal-dual qui converge pour des problèmes convexes à condition que $\tau, \sigma > 0$ et $\tau\sigma < \frac{1}{16}$ (voir Section ci-après 3.2.4).

Algorithme 4 Algorithme primal dual appliqué au problème non convexe.

$Z \leftarrow 0$
for $i = 1$:itération **do**
  $Z \leftarrow PB\left(Z + \sigma\nabla u\right)$
  $W \leftarrow P_\Delta\left(W - \tau_w((||u - c_i||^2)_i + \alpha(1-2W)\right)$
  $u \leftarrow PA\left(\frac{u + \tau\left(\mathrm{div}(Z) + \lambda\sum_i w_i c_i\right)}{1+\tau\lambda}\right)$
end for

où P_Δ est la projection sur le simplexe $\{(w_1,\ldots,w_n)$ tel que $0 \leqslant w_i \leqslant 1\ , \forall i \in [1..n]$ et $\sum_i w_i = 1\}$, donné par [31]. PA est la projection sur le carré $[0,255]^2$. PB est la projection sur la boule unité $\sqrt{x^2+y^2} \leqslant 1$.

On peut observer le résultat sur un premier problème jouet de colorisation simple. On utilise pour image cible le canal de luminance de l'image source. De cette manière, on s'affranchit d'une partie des problèmes de texture. On laisse converger l'algorithme (Figure 8).

(a) Image source. (b) Convergence du modèle de Bugeau *et al.* [1].

FIGURE 8 – auto-colorisation par le modèle Bugeau *et al.* [1]. On s'affranchit ici du problème de la recherche des candidats afin de connaitre l'effet de la régularisation.

On remarque que le résultat de l'algorithme est terne. Notons que cela provient du modèle en lui-même. En effet, si l'on imagine que l'on régularise très fortement avec ce modèle, à Y (canal de luminance) fixé, les canaux U et V de chrominance auront tendance à suivre une distribution constante. Et donc on limitera beaucoup la palette de couleurs disponibles. On peut vérifier cela de manière qualitative : pour un niveau de luminance donné on trace les points RGB qui lui appartiennent en prenant trois points du polygone constitué de l'intersection du plan de luminance constante avec le cube RGB, avec pour repère, le repère cartésien naturel (voir Figure 9). On trace ces nuages de points

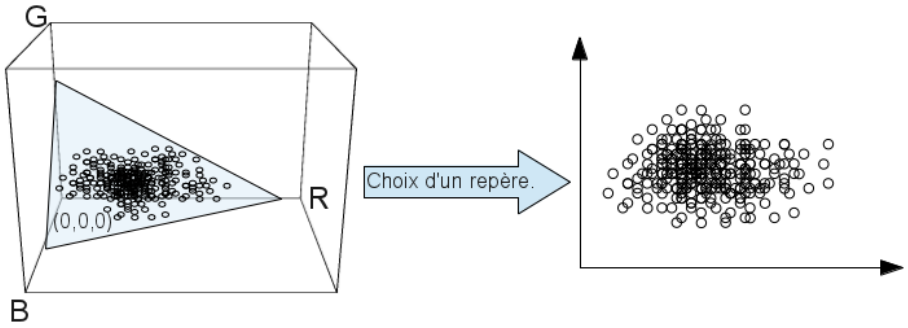
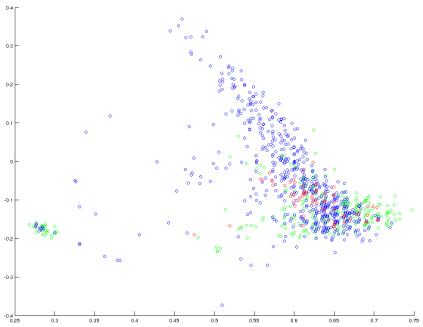


FIGURE 9 – Pour un niveau de luminance donné on trace les points RGB qui lui appartiennent en prenant trois points du polygone constitué de l'intersection du plan de luminance constante avec le cube RGB.

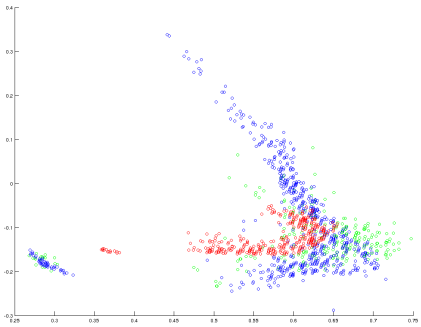
(Figure 10). On vérifie qu'il y a bien un resserrement du modèle YUV vers des valeurs centrales ternes lors de la convergence de l'algorithme.

On observe en ronds rouges, les couleurs du modèle de Bugeau *et al.*, en ronds verts, les couleur de l'image initiale. À l'initialisation, on observe que les deux nuages ont un resserrement équivalent. En revanche, ce que l'on avait prévu, à savoir le resserrement du nuage vers un point (distribution constante) se réalise. Et ce resserrement se fait vers la zone du plan contenant des couleurs grises. Cela montre que l'impression d'image terne provient bien du modèle en lui-même. Il semblerait donc qu'il faille rejeter la régularisation par minimisation de la variation totale sur les canaux de chrominance.

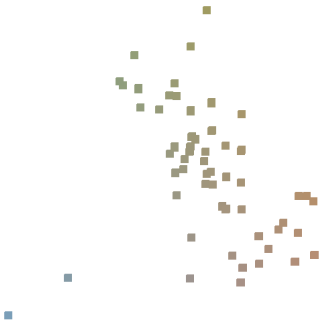
On remarque aussi un phénomène de flou au niveau des contours de l'image, par exemple au niveau de la limite entre la montagne et le ciel (figure 8(b)). Ce phénomène est surprenant au premier abord, puisque l'espace des fonctions à variations bornées (*i.e.* l'espace des fonctions intégrables de variation totale finie) peut admettre des discontinuités. En fait l'explication est simple : la définition de la variation totale de Bugeau *et al.* ne prend en compte que des canaux U et V, et donc ne sont couplés que les canaux de chrominance, sans l'être avec le canal de luminance. La minimisation d'une telle variation totale permet donc une distinction des contours dans les espaces de chrominance et sur l'espace de luminance. Ainsi, la colorisation n'est pas suffisamment liée à l'image en niveaux de gris cible. On rejettera donc définitivement la variation totale sur les canaux de chrominance.



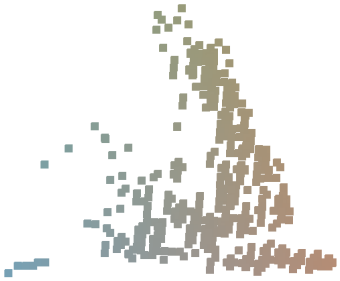
(a) Nuage de points à l'initialisation.



(b) Nuage de points à convergence.



(c) Couleurs correspondant aux points du nuage à l'initialisation du modèle Bugeau *et al.*



(d) Couleurs correspondant aux points du nuage à convergence du modèle Bugeau *et al.*

FIGURE 10 – Nuages de points pour une luminance donnée et la carte des couleurs correspondante. On obtient ces nuages en traçant à plat l'ensemble des points de l'espace RGB correspondants aux couleurs d'une image dont la luminance a un niveau fixé : ces points sont effectivement coplanaires dans l'espace RGB. En rouge, le nuage correspondant au modèle de Bugeau *et al.* [1], en bleu, notre modèle, en noir, l'image initiale qui sert de vérité. À droite, les couleurs correspondant au nuage rouge (modèle de Bugeau *et al.*) : remarquons que les valeurs se resserrent vers le gris.

3 Évolution du modèle et contributions.

3.1 Évolution du modèle vers l'espace RGB.

Le fil directeur de cette partie sera l'évolution et l'amélioration du modèle proposé par Bugeau, Ta et Papadakis [1]. Une première évolution sera de se placer (on verra l'intérêt en Section 3.3.3) dans l'espace RGB au lieu de l'espace YUV. On s'attachera aussi à modifier l'attache aux données.

3.1.1 Inconvénients et avantages de l'espace YUV dans le problème de la colorisation basée-exemple.

Comme expliqué dans la section précédente, l'utilisation de l'espace de luminance-chrominance YUV présente certains inconvénients : resserrement des valeurs et apparition de gris, entrainant à l'apparition d'images ternes, manque de couplage avec la luminosité, et manque de prise en compte de la luminosité dans le modèle d'inférence (sélection du meilleur candidat). Ce dernier point est problématique. En effet, soient U et V deux canaux de chrominance fixés, en faisant varier la luminance, on peut obtenir beaucoup de teintes possibles (voir figure 11).

FIGURE 11 – Si U et V sont constants et Y varie, différentes teintes, non proportionnelles peuvent être atteintes.

En revanche, cette recherche possède un avantage non négligeable : cela donne une métrique invariante vis-à-vis de la différence d'éclairage entre les deux scènes. Donc on va modifier notre modèle pour se placer en RGB, et sans tenir compte de la luminosité lors de la recherche des patchs. La conservation de la luminosité sera obtenue grâce à l'ajout d'une contrainte dans la fonctionnelle. Ainsi, on garde les avantages du modèle de Bugeau *et al.* [1] sans avoir les inconvénients mentionnés en figure 11.

3.1.2 Étude qualitative et comparative des fonctionnelles.

Les résultats du modèle Bugeau *et al.* sont déjà corrects, mais on peut noter quelques défauts de ce modèle dont on aimerait s'affranchir afin de le rendre plus évolutif par la suite. Tout d'abord, le défaut majeur de ce modèle, qui est immédiatement apparent, est que l'espace YUV ne se prête pas à l'utilisation de la variation totale, puisque si l'on régularise trop, on fait tendre les canaux U et V vers une distribution constante, et cela rend les images ternes. De plus, le défaut du découplage luminance-chrominance au niveau des contours est rédhibitoire. On utilisera donc la variation totale sur les trois canaux RGB. Afin de donner un sens à la variation totale sur un modèle à trois primitives couleur, on aimerait pouvoir utiliser la variation totale adaptée aux images couleurs introduite par Goldluecke *et al.* [25]. Ensuite, le problème de l'invariance du choix des patchs vis-à-vis de la luminance, citée plus haut doit être résolu. On est donc tenté pour cela de se placer dans l'espace à trois primitives RGB. De plus, afin de pouvoir avoir une attache aux données différente de la norme euclidienne sur les canaux de chrominance, il peut-être plus facile de travailler à partir de RGB car tous les systèmes de couleurs sont définis à partir de RGB. Enfin, le dernier argument en faveur de l'espace RGB, est l'absence de transformation en entrée et en sortie de l'algorithme, ce qui est problématique pour certains espaces de couleur qui ne sont pas toujours convertibles.

Afin de prendre en compte tous les éléments mentionnés, on pose donc la fonctionnelle suivante, u étant une image RGB :

$$\begin{aligned} F(u,W) := \quad & TV(u) + \frac{\lambda}{2}\int_\Omega \sum_{i=1}^{8} w_i ||A\,(u-c_i)\,||^2 + \frac{\alpha}{2}\int_\Omega \sum_{i=1}^{8} w_i(1-w_i) \\ & + \chi_{u\in[0,255]} + \chi_{Y(u)=I_g} + \chi_{W\in\Delta}. \end{aligned} \tag{53}$$

Le terme $\int_\Omega \sum_{i=1}^8 w_i ||A\,(u-c_i)\,||^2$ est celui qui permet de faire le lien entre les couleurs candidates (c_i) et la couleur qui sera effectivement retenue (u). A est une matrice inversible qui permet de passer de l'espace RGB à un espace différent. On pourra prendre $A = I_d$ la matrice identité pour avoir une attache aux données en norme euclidienne sur les canaux RGB, mais on pourra essayer d'autres espaces couleurs en prenant pour A une des matrices transformant les coordonnées de l'espace RGB vers d'autres espaces couleurs (matrices données en exemple dans la Section 2.2). Les diverses couleurs sont pondérées par les poids $1 \leqslant w_i \leqslant 1$, et on pourrait ainsi converger vers des couleurs qui seraient des mélanges de deux teintes, mais ne seraient pas des couleurs plausibles vis-à-vis des couleurs de l'image source. Par exemple, la moyenne du rouge et du vert est un jaune pâle qui n'a visuellement rien à voir avec les couleurs initiales. Il faut donc éviter cela en forçant les couleurs à converger vers un unique candidat plutôt qu'une moyenne de plusieurs couleurs. Pour cela on introduit le terme $\dfrac{\alpha}{2}\displaystyle\int_\Omega \sum_{i=1}^8 w_i(1-w_i)$ qui va forcer les coefficients à prendre pour valeurs 0 ou 1. Néanmoins, pour avoir une vraie moyenne (et éviter, par exemple de se retrouver avec deux valeurs valant 1), on ajoute la fonction $\chi_{W\in\Delta}$ qui vaut $+\infty$ si la contrainte $W \in \Delta$ n'est pas respectée, et 0 sinon. $W \in \Delta$ signifie que pour chaque pixel, le vecteur $(w_1, \ldots, w_8)$ vérifie les propriétés, $0 \leqslant w_i \leqslant 1$, $\forall i \in [1..8]$ et $\sum_{i=1}^8 w_i = 1$. On appellera cette contrainte, dans la suite, être sur le simplexe.

Posée ainsi, la fonctionnelle n'est pas convexe, et admet beaucoup de minima locaux. De plus, elle ne permet pas d'avoir une régularité spatiale de l'image réalisant le minimum. En ajoutant ce terme $TV(u)$ on permet cette régularisation. En effet, il est minimal quand l'image est constante : la variation totale permet d'avoir des fonctions constantes par morceaux, ce qui est un modèle variationnel relativement proche de la vision humaine [5]. Enfin on ajoute deux contraintes supplémentaires : la première est d'avoir un niveau de gris égal à celui de l'image cible, ce qui est indispensable pour conserver les informations texturales, la deuxième est de rester sur le cube RGB, c'est-à-dire entre 0 et 255 pour chaque canal de couleur. La contrainte de gris $Y(u) = I_g$ est donnée par $A.u = I_g$ où u est le vecteur à trois composantes, R, G et B, I_g est un réel désignant la luminance de l'image cible, et $A = (0.2990, 0.5870, 0.1140)$.

3.2 Résolution par algorithme primal-dual.

3.2.1 Positionnement du problème.

On veut minimiser la fonctionnelle (53), pour rappel :

$$\begin{aligned} F(u,W) := \quad & TV(u) + \frac{\lambda}{2}\int_\Omega \sum_{i=1}^8 w_i ||A\,(u-c_i)\,||^2 + \frac{\alpha}{2}\int_\Omega \sum_{i=1}^8 w_i(1-w_i) \\ & + \chi_{u\in[0,255]} + \chi_{Y(u)=I_g} + \chi_{W\in\Delta}. \end{aligned} \tag{54}$$

Pour la minimiser par rapport à u, il est possible d'appliquer un algorithme primal-dual. Pour cela il faut le problème dual (on fait abstraction des termes dépendant de W) :

$$\begin{aligned} \min_u \max_p -\langle \operatorname{div} p | u\rangle + \quad & \frac{\lambda}{2}\int_\Omega \sum_{i=1}^8 w_i ||A\,(u-c_i)\,||^2 + \frac{\alpha}{2}\int_\Omega \sum_{i=1}^8 w_i(1-w_i) \\ & + \chi_{u\in[0,255]} + \chi_{Y(u)=I_g} + \chi_{\sqrt{z_1^2+z_2^2}\leqslant 1}\,(p)\,. \end{aligned} \tag{55}$$

3.2.2 Le problème de la luminance et des bornes.

Le problème qui apparait naturellement quand on veut mettre en place l'algorithme primal-dual est le calcul de la projection de u sur la contrainte, *i.e.*, le calcul de l'opérateur

5. On pourra à l'avenir changer cette régularisation et plutôt mettre un terme de variation totale adapté à la couleur ou encore une variation totale généralisée.

proximal de $\chi_{u\in[0,255]} + \chi_{Y(u)=I_g}$. Il est donc nécessaire d'être capable de calculer cette projection pour mettre en place l'algorithme primal-dual. La mise en place de l'algorithme primal-dual nous amène donc naturellement à trouver un moyen de calculer la projection exacte sur l'ensemble convexe correspondant à la contrainte :

$$\chi_{u\in[0,255]} + \chi_{Y(u)=I_g}.$$

Cela revient à déterminer la projection sur l'intersection du cube (*cube RGB*) correspondant à la contrainte $\chi_{u\in[0,255]}$ et le plan affine correspondant à la contrainte $\chi_{Y(u)=I_g}$. Ceci correspond soit à un singleton soit à l'enveloppe convexe de trois, quatre ou cinq points de l'espace ni confondus, ni alignés, mais coplanaires.

Pour projeter on procède en décomposant de la manière suivante : on commence par projeter sur le plan (voir Lemme 3.3) correspondant à la contrainte puis on projette sur l'intersection. Ceci est parfaitement justifié par la théorie.

Proposition 3.1 (Caractérisation de la projection.)**.** *Soit H un espace de Hilbert, et $u \in H$, soit C un ensemble fermé convexe non vide inclus dans H, soit $v \in H$.*

Alors v est la projection orthogonale de u sur C si et seulement si

$$\langle u - v | z - v \rangle \leqslant 0 \ , \ \forall z \in C \ ,$$

Lemme 3.2. *Soit $x \in \mathbb{R}^3$, et P un plan de $\mathbb{R}^3$ affine, u la projection orthogonale de x sur P. Soit C un ensemble fermé convexe non vide inclus dans P. Et soit v la projection orthogonale de u sur C.*

Alors v est la projection orthogonale de x sur C.

Démonstration. On écrit (caractérisation de la projection),

$$\langle u - v | z - v \rangle \leqslant 0 \ , \ \forall z \in C \ ,$$

et

$$\langle x - u | v - u \rangle = 0 \ ,$$

car la projection est orthogonale.

On veut montrer que $\langle x - v | z - v \rangle \leqslant 0 \ , \ \forall z \in C$. Écrivons

$$\langle x - u + u - v | z - v \rangle = \langle x - u | z - v \rangle + \langle u - v | z - v \rangle \, .$$

D'une part $\langle x - u | z - v \rangle = 0$ car $z \in P$, puisque $C \subset P$. D'autre part $\langle u - v | z - v \rangle \leqslant 0$ par hypothèse. On a bien montré que l'on peut composer les projections.

□

Donnons maintenant la projection d'un point sur un plan.

Lemme 3.3. *Soit $a \in \mathbb{R}^n$, $a \neq 0$, soit $W := \{x \in \mathbb{R}^n \text{ tel que } \langle a | x \rangle = \alpha\}$, et soit $s \in \mathbb{R}^n$.*

Alors la projection orthogonale de s sur l'hyperplan affine W est donnée par :

$$P_W(s) = \frac{a}{||a||_2} \left(\frac{\alpha - \langle a | s \rangle}{||a||_2} \right) + s. \tag{56}$$

Démonstration. On fait les manipulations algébriques suivantes :

$$\begin{aligned} W &:= \{x \in \mathbb{R}^n \text{ tel que } \langle a | x \rangle = \alpha\} \\ &= \left\{ x \in \mathbb{R}^n \text{ tel que } \left\langle \frac{a}{||a||_2} | x \right\rangle = \frac{\alpha}{||a||_2} \right\} \\ &= \left\{ x \in \mathbb{R}^n \text{ tel que } \left\langle \frac{a}{||a||_2} | x - \alpha \frac{a}{||a||_2} \right\rangle = 0 \right\} \\ &= \alpha \frac{a}{||a||_2} + \left\{ y \in \mathbb{R}^n \text{ tel que } \left\langle \frac{a}{||a||_2} | y \right\rangle = 0 \right\} \\ &= \alpha \frac{a}{||a||_2} + \left(\overrightarrow{\text{vect}} \left(\frac{a}{||a||_2} \right) \right)^{\perp}. \end{aligned} \tag{57}$$

Ainsi :

$$\begin{aligned} P_W &= \alpha\frac{a}{||a||_2} + P_{\left(\overrightarrow{\text{vect}}\left(\frac{a}{||a||_2}\right)\right)^{\perp}}(s) \\ &= \alpha\frac{a}{||a||_2} + \left(s - P_{\overrightarrow{\text{vect}}\left(\frac{a}{||a||_2}\right)}(s)\right) \\ &= \alpha\frac{a}{||a||_2} + \left(s - \left\langle \frac{a}{||a||_2} | s \right\rangle \frac{a}{||a||_2}\right). \end{aligned} \tag{58}$$

Au final :

$$P_W(s) = \frac{a}{||a||_2}\left(\frac{\alpha - \langle a|s\rangle}{||a||_2}\right) + s. \tag{59}$$

□

La projection sur le plan (cf. Lemme 3.3) est donc donnée par l'algorithme 5.

Algorithme 5 Projection sur le plan $AX = I_g$.

1: $X \leftarrow \dfrac{A}{||A||^2}\left(I_g - \langle X|A\rangle\right) + X$

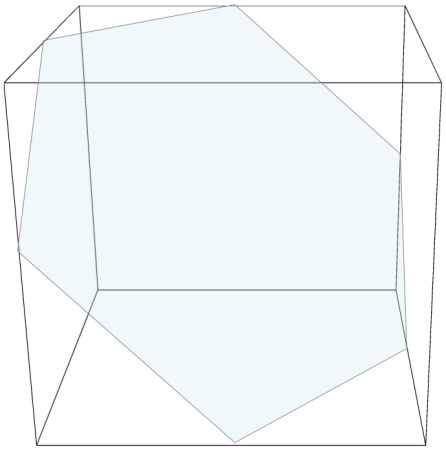


FIGURE 12 – L'intersection d'un cube et d'un plan peut être un point, un segment, un triangle, un quadrilatère, un pentagone, un hexagone ou l'ensemble vide.

Ceci nous ramène à reconsidérer le problème dans le plan. Soit 3, 4, 5 ou 6 points du plan en configuration naturelle [6] (voir figure 12), et soit un point X dont on cherche la projection sur le polygone constitué par ces points. Il y a deux possibilités : soit X est déjà dans l'enveloppe convexe de ces points (ce que j'appellerai désormais le polygone) soit X est à l'extérieur de ces points. Dans ce dernier cas, la projection se trouvera sur un des segments constituant le bord de ce polygone.

Maintenant que l'on a projeté le point sur le plan, puisque le polygone est sur le plan, il faut projeter sur le polygone. Il y a deux possibilités : soit le point est déjà dans ce polygone, soit le point est hors de ce polygone.

6. Dont aucun point n'est dans l'enveloppe convexe des autres points. Notons également que dans notre cas, on n'atteindra jamais l'hexagone.

Il faut donc mettre au point deux premiers algorithmes, un qui décide si le point considéré est dans le polygone, et un qui projette sur un segment défini par deux points donnés.

Si A et B sont deux points de l'espace, on calcule P la projection de X sur le segment $[A, B]$, donnée par l'algorithme 6.

Algorithme 6 Projection sur le segment $[A, B]$.

$\lambda \leftarrow \dfrac{\langle X - A | B - A \rangle}{||B - A||^2}$
if $\lambda \geqslant 1$ **then**
    $P \leftarrow B$
else if $\lambda \leqslant 0$ **then**
    $P \leftarrow A$
else
    $P \leftarrow A + \lambda(B - A)$
end if

Pour savoir si le point X est dans le polygone, on réfléchit comme suit : on a projeté sur le plan, donc le point à projeter appartient au plan. Il appartient au cube si et seulement si il appartient au polygone. Il suffit donc de vérifier deux inégalités par coordonnées soit 6 tests d'inégalité au total. Au final, cela nous permet d'avoir l'opérateur proximal pour la fonction $\chi_{U \in [0,255]} + \chi_{Y(U)=I_g}$. J'appellerai dans la suite PG_{I_g} cette projection. On peut, grâce à cela former un opérateur proximal pour cette fonctionnelle.

3.2.3 Mise en place à proprement parler de l'algorithme primal-dual.

Maintenant que l'on dispose de cette projection on peut mettre en place l'algorithme primal-dual pour minimiser la fonctionnelle 54 (voir algorithme 2) : on identifie $F^*(Z) = \chi_{\max(\sqrt{z_1^2+z_2^2},1)}(z)$, $Ku = \nabla u$, et

$$G(u) = \frac{\lambda}{2} \int_\Omega \sum_{i=1}^{8} w_i ||A(u - c_i)||^2 + \chi_{u \in [0,255]} + \chi_{Y(u)=I_g}.$$

Il faut maintenant calculer les opérateurs proximaux de G et F^*. Celui de F^* est déjà connu [22], et est donné par une projection :

$$\text{prox}_{\sigma F^*}(p) = (I + \sigma \partial F^*)^{-1}(p) = \begin{cases} p_{i,j} & \text{si } ||p_{i,j}||_2 \leqslant 1 \\ \dfrac{p_{i,j}}{||p_{i,j}||_2} & \text{sinon.} \end{cases} \tag{60}$$

Pour G, on se place dans le cas particulier de $A = I_d$. Pour calculer l'opérateur proximal, on revient à sa définition :

$$\text{prox}_{\tau G}(\tilde{u}) \quad = \text{argmin}_u \int_\Omega \frac{||u - \tilde{u}||^2}{2\tau} + \frac{\lambda}{2} \int_\Omega \sum_{i=1}^{8} w_i ||u - c_i||^2 + \chi_{u \in [0,255]} + \chi_{Y(u)=I_g}. \tag{61}$$

La fonctionnelle étant séparable (pixel par pixel), on omet $\int_\Omega$:

$$
\begin{aligned}
\text{prox}_{\tau G}(\tilde{u}) &= \text{argmin}_u \frac{||u-\tilde{u}||^2}{2\tau} + \frac{\lambda}{2}\sum_{i=1}^{8} w_i||u-c_i||^2 + \chi_{u\in[0,255]} + \chi_{Y(u)=I_g} \\
&= \text{argmin}_u \frac{\langle u|u\rangle - 2\langle u|\tilde{u}\rangle + \langle \tilde{u}|\tilde{u}\rangle}{2\tau} + \frac{\lambda}{2}\sum_{i=1}^{8} w_i\left[\langle u|u\rangle - 2\langle u|c_i\rangle + \langle c_i|c_i\rangle\right] \\
&+\chi_{u\in[0,255]} + \chi_{Y(u)=I_g} \\
&= \text{argmin}_u \frac{1+\lambda\tau}{2\tau}\langle u|u\rangle - \frac{2}{2\tau}\left\langle u|\tilde{u}+\tau\lambda\textstyle\sum_{i=1}^{8} w_i c_i\right\rangle - \frac{||\tilde{u}||^2}{2\tau} + \frac{\lambda\sum w_i||c_i||^2}{2} \\
&+\chi_{u\in[0,255]} + \chi_{Y(u)=I_g} \\
&= \text{argmin}_u \frac{1+\lambda\tau}{2\tau}\left[\langle u|u\rangle - \frac{1}{1+\lambda\tau}\left\langle u|\tilde{u}+\tau\lambda\textstyle\sum_{i=1}^{8} w_i c_i\right\rangle\right. \\
&\left. + \left(\frac{2\tau}{1+\lambda\tau}\right)^2 ||\tilde{u}+\tau\lambda\textstyle\sum_{i=1}^{8} w_i c_i||^2\right] - \left(\frac{2\tau}{1+\lambda\tau}\right)||\tilde{u}+\tau\lambda\textstyle\sum_{i=1}^{8} w_i c_i||^2 \\
&-\frac{||\tilde{u}||^2}{2\tau} + \frac{\lambda\sum w_i||c_i||^2}{2} + \chi_{u\in[0,255]} + \chi_{Y(u)=I_g} \\
&= \text{argmin}_u \left\| u - \frac{\tilde{u}+\tau\lambda\sum_{i=1}^{8} w_i c_i - u}{1+\tau\lambda}\right\|^2 \\
&+\chi_{u\in[0,255]} + \chi_{Y(u)=I_g}.
\end{aligned}
\tag{62}
$$

Ce qui correspond à la projection de $\frac{\tilde{u}+\tau\lambda\sum_{i=1}^{8} w_i c_i}{1+\lambda\tau}$ sur la contrainte $\chi_{u\in[0,255]} + \chi_{Y(u)=I_g} < +\infty$.

Pour la cas général de $A \in \mathcal{M}_3(\mathbb{R})$, on écrit l'équation à point fixe à partir d'un schéma explicite :

$$
u_{n+1} = \text{argmin}_{\tilde{u}} \frac{||u_n-\tilde{u}||_2^2}{2\delta} + \frac{\lambda}{2}\sum_{i=1}^{8} w_i||A\,(u_n-c_i)\,||_2^2. \tag{63}
$$

Ou encore, à condition que $I-\delta\lambda A^tA$ soit inversible :

$$
u_{n+1} = \left(I-\delta\lambda A^tA\right)^{-1}\left(u_n + \delta\lambda A\sum_{i=1}^{8} w_i c_i\right). \tag{64}
$$

Ainsi, dans le cas où $A = Id$ on donne un algorithme pour minimiser en u la fonctionnelle (54) par optimisation de (55), dont on sait qu'il converge.

On appellera PG la projection sur la contrainte $\chi_{u\in[0,255]} + \chi_{Y(u)=I_g} < +\infty$.

On appellera PB l'opérateur proximal de $\chi_{\sqrt{z_1^2+z_2^2+\cdots+z_6^2}\leqslant 1}(z)$, ou la projection donnée par l'algorithme 3 dans le cas de la variation totale vectorisée.

L'algorithme primal-dual associé à ce problème est l'algorithme 7 :

Algorithme 7 Algorithme primal dual minimisant la fonctionnelle par rapport à u dans le cas particulier $A = Id$.

1: $Z \leftarrow 0$
2: **for** $i \leqslant n$ **do**
3: $\quad Z \leftarrow PB\,(Z + \sigma\nabla u)$
4: $\quad u \leftarrow PG\left(\frac{u+\tau\,(\text{div}(Z)+\lambda\sum_i w_i c_i)}{1+\tau\lambda}\right)$
5: **end for**

L'avantage de cet algorithme est que l'on sait que, pour des paramètres bien choisis ($\tau, \sigma > 0$ et $\tau\sigma < \frac{1}{24}$, voir section 3.2.4), on obtient convergence, car le problème est convexe.

Dans le cas général $A \in \mathcal{M}_3(\mathbb{R})$, on écrit l'algorithme 8.

Algorithme 8 Algorithme primal dual minimisant la fonctionnelle par rapport à u dans le cas général.

1: $Z \leftarrow 0$
2: **for** $i = 1$:itération **do**
3: $\quad Z \leftarrow PB\left(Z + \sigma \nabla u\right)$
4: $\quad u \leftarrow PG\left(\left(I - \delta\lambda A^t A\right)^{-1}\left(u + \tau\left(\mathrm{div}(Z) + \lambda \sum_i w_i c_i\right)\right)\right)$
5: **end for**

3.2.4 Étude des opérateurs.

On reprend les définitions des gradients et divergences discrètes pour les images couleurs :

Définition 3.4. Posons

$$\nabla u_{i,j} = \begin{pmatrix} \begin{cases} u_{i+1,j,1} - u_{i,j,1} \text{ si } i < N, \\ 0 \text{ si } i = N, \end{cases} \\ \begin{cases} u_{i,j+1,1} - u_{i,j,1} \text{ si } j < N, \\ 0 \text{ si } j = N. \end{cases} \\ \vdots \\ \begin{cases} u_{i,j+1,3} - u_{i,j,3} \text{ si } j < N, \\ 0 \text{ si } j = N. \end{cases} \end{pmatrix}. \tag{65}$$

On pose également :

$$\begin{aligned} div(p)_{i,j} \quad &= \begin{cases} p^1_{i,j} - p^1_{i-1,j} \text{ si } 1 < i < N, \\ p^1_{i,j} \text{ si } i = 1, \\ -p^1_{i-1,j} \text{ si } i = N, \end{cases} + \begin{cases} p^2_{i,j} - p^2_{i,j-1} \text{ si } 1 < j < N, \\ p^2_{i,j} \text{ si } j = 1, \\ -p^2_{i,j-1} \text{ si } j = N, \end{cases} \\ &+ \cdots \\ &+ \begin{cases} p^5_{i,j} - p^5_{i-1,j} \text{ si } 1 < i < N, \\ p^5_{i,j} \text{ si } i = 1, \\ -p^5_{i-1,j} \text{ si } i = N, \end{cases} + \begin{cases} p^6_{i,j} - p^6_{i,j-1} \text{ si } 1 < j < N, \\ p^6_{i,j} \text{ si } j = 1, \\ -p^6_{i,j-1} \text{ si } j = N, \end{cases} . \end{aligned} \tag{66}$$

En dimension 1 (1 canal, l'image est évidemment en dimension 2) le carré de la norme de l'opérateur divergence est, en négligeant les effets de bords, égale à 8 (voir [21], Remarque page 92.) La convergence de cet algorithme vers un minimiseur est donc vérifiée pour $\tau, \sigma > 0$ tels que $\tau\sigma < \frac{1}{8}$ (Voir [22], Théorème 1.)

Dans le cas de trois canaux de couleur, en négligeant les effets de bords, et en simplifiant (pour les notations) à une image carrée :

$$\begin{aligned} ||\mathrm{div} p||_2^2 \quad &= \sum_{1 \leqslant i,j \leqslant N} \left(p^1_{i,j} - p^1_{i-1,j} + p^2_{i,j} - p^2_{i,j-1} + \cdots + p^6_{i,j} - p^6_{i,j-1}\right)^2 \\ &\leqslant 12 \sum_{1 \leqslant i,j \leqslant N} \left(p^1_{i,j}\right)^2 + \left(p^1_{i-1,j}\right)^2 + \left(p^2_{i,j}\right)^2 + \left(p^2_{i,j-1}\right)^2 + \cdots + \left(p^6_{i,j}\right)^2 + \left(p^6_{i,j-1}\right)^2 \\ &\leqslant 24 ||p||_2^2 \end{aligned} \tag{67}$$

En choisissant $p^1_{i,j} = p^2_{i,j} = \cdots = p^6_{i,j} = (-1)^{i+j}$, on obtient que $\kappa = |||\mathrm{div}||| = \sup_{||p|| \leqslant 1} ||\mathrm{div} p||_2$, la norme de l'opérateur divergence s'approxime (effets de bords) $\kappa = 24 - O(N)$, avec $N =$ largeur + hauteur en pixels. On utilisera la valeur 24 pour le carré de la norme de l'opérateur div, égal, par le Théorème 2.15, à celui de l'opérateur gradient.

3.2.5 Mise en place dans le cas non-convexe.

Pour minimiser aussi la fonctionnelle par rapport à u et W on ajoute une ligne de type gradient projeté dans les itérations de l'algorithme, ce qui donne l'algorithme 9.

Algorithme 9 Algorithme primal dual appliqué au problème non convexe.

$Z \leftarrow 0$
for $i = 1$:itération **do**
  $Z \leftarrow PB\left(Z + \sigma \nabla u\right)$
  $W \leftarrow P_\Delta\left(W - \tau_w((||u - c_i||^2)_i + \alpha(1 - 2W)\right)$
  $u \leftarrow PG\left(\left(I - \delta\lambda A^t A\right)^{-1}\left(u + \tau\left(\mathrm{div}(Z) + \lambda \sum_i w_i c_i\right)\right)\right)$
end for

Où P_Δ est la projection sur le simplexe $\{(w_1, \ldots, w_n)$ tel que $0 \leqslant w_i \leqslant 1$, $\forall i \in [1..n]$ et $\sum_i w_i = 1\}$, donné par [31]. u et W sont les variables primales et Z est la variable duale.

3.3 Résultats numériques.

3.3.1 Observation des résultats.

(a) Source de beach. (b) Cible beach. (c) Source de paysage. (d) Cible paysage.

(e) Beach colorisée. (f) Paysage colorisée.

FIGURE 13 – Résultat de l'algorithme primal-dual appliqué au modèle RGB avec régularisation par minimisation de la variation totale. Les trois canaux de couleur sont couplés en position et direction.

La figure 13 présente deux résultats de colorisation. On remarque qu'il n'y a pas d'artefacts. Les couleurs sont vives, respectent une cohérence visuelle. Néanmoins, sur l'image paysage, il reste une tâche marron en haut à droite qui est rédhibitoire. Il faudra l'éliminer à l'avenir. Néanmoins celle-ci reste moins visible qu'avec l'algorithme de Forward-Backward généralisé, que nous avons éprouvé (Section A.1).

Deux autres résultats obtenus sur une image de texture similaire à la source et sur une image ancienne sont présentés sur les figures 14 et (figure 15).

Image non libre de droit.

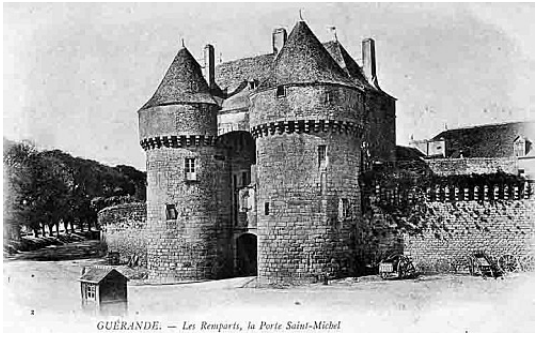
Image non libre de droit.

Image non libre de droit.

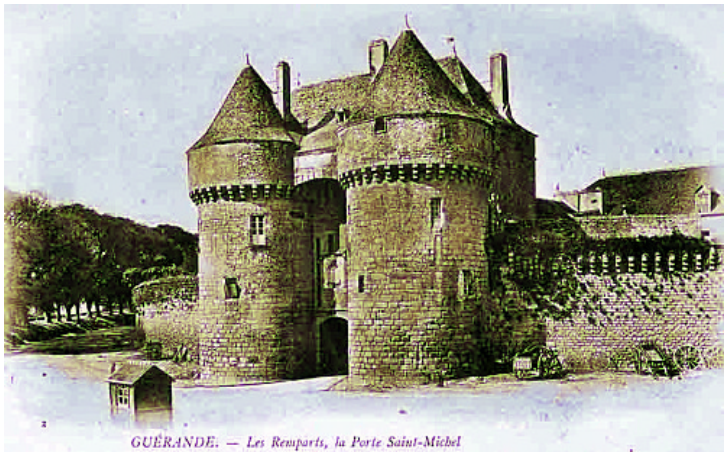
(a) Source. (b) Cible. (c) Image colorisée.

FIGURE 14 – L'image source et cible formant initialement une seule et même image, cela permet de conserver des propriétés similaires au niveau des textures pour la source et la cible.

(a) Image source, photo actuelle.



(b) Image cible, ancienne.

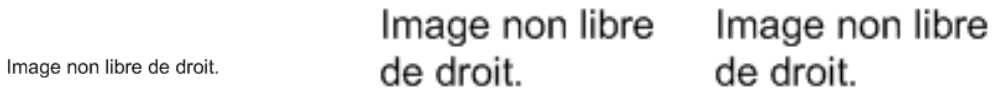


(c) Image colorisée.

FIGURE 15 – Il s'agit de la colorisation d'une image ancienne. L'image cible semble bien colorisée, et l'algorithme semble donner de bons résultats, mais ce n'est pas réellement le cas. On utilisera cette image pour montrer les problèmes que l'on a à l'initialisation (Section 3.5.2).

Nous avons ensuite testé l'algorithme 9 sur des images particulières : les portraits

(figure 16). Chen *et al.* [14] font remarquer que la colorisation des visages est un sujet de difficulté majeure sur lesquels les algorithmes de colorisation ont des difficultés à fonctionner. Cela vient pour l'essentiel du fait que la peau et les visages contiennent beaucoup de parties lisses, et ces parties sont toujours difficiles à coloriser par manque d'information à cause d'un manque de textures. On remarque que globalement le coté lisse du visage, le grain de la peau et son teint sont naturels et ne semblent pas provenir d'un algorithme de colorisation. Également la zone des cheveux, bien texturée, est bien colorisée. En revanche, la bouche et les yeux sont mal colorisés. On remarque aussi que cet algorithme fonctionne bien grâce à des images d'entrées particulièrement bien choisies pour cela : il n'y a que la peau du visage qui soit lisse sur chacune des images source et cible, ce qui permet à l'algorithme de ne pas faire d'erreurs par confusion de zones lisses.



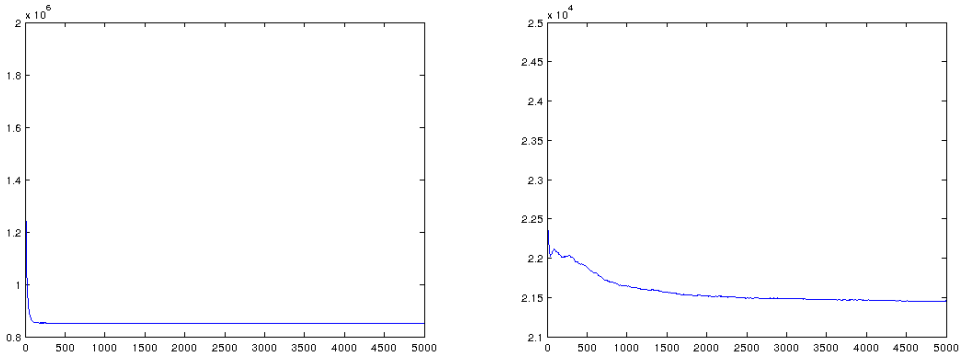
(a) Image source, Maria Sharapova. (b) Image cible, Caroline Wozniacki. (c) Image colorisée.

Crédit photographique pour l'image Sharapova : © Istockphotos.
Crédit photographique pour l'image Wozniacki : © getty.

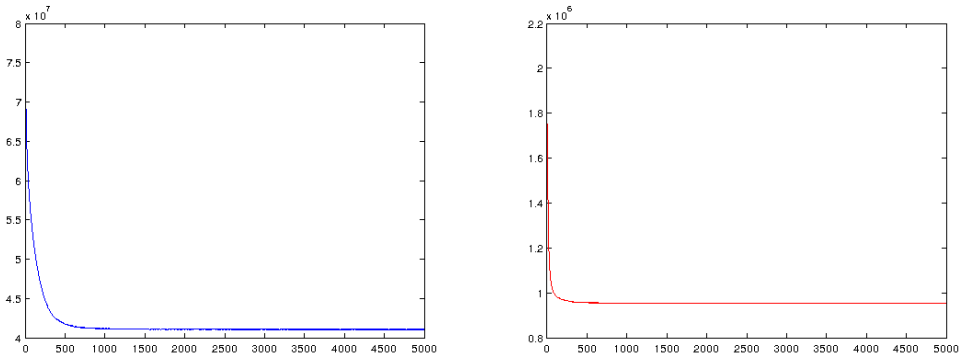
FIGURE 16 – Colorisation de visages.

3.3.2 Convergence empirique de l'algorithme.

Rappelons que le modèle de départ était non-convexe. Il était convexe en la variable u mais pas en W. L'algorithme primal-dual mis en place converge dans le cas convexe [22] , mais il n'existe pas à notre connaissance de preuve de convergence dans le cas non convexe, même si les figures 39 nous laissent facilement conjecturer que la convergence a lieu. On peut constater la convergence de l'algorithme via la convergence de l'énergie. Si l'on note $(u_n, W_n)_{n\in\mathbb{N}}$ les itérés de l'algorithme, on peut observer que la courbe de $F(u_n, W_n)_{n\in\mathbb{N}}$ semble converger (figure 17).



(a) $TV(u)$ (b) $\int_\Omega \sum_{i=1}^{8} w_i(1-w_i)$



(c) $\int_\Omega \sum_{i=1}^{8} w_i ||A\,(u-c_i)\,||^2$ (d) $= (a) + \frac{\lambda}{2}(b) + \frac{\alpha}{2}(c)$

FIGURE 17 – Valeur de la fonctionnelle en fonction du nombre d'itérations. On observe une décroissance de la fonctionnelle lors de la convergence de l'algorithme primal-dual appliqué au modèle RGB. L'algorithme semble converger en pratique bien que nous n'ayons pas de preuve théorique.

3.3.3 Analyse et comparaison avec les résultats de Bugeau *et al.* [1].

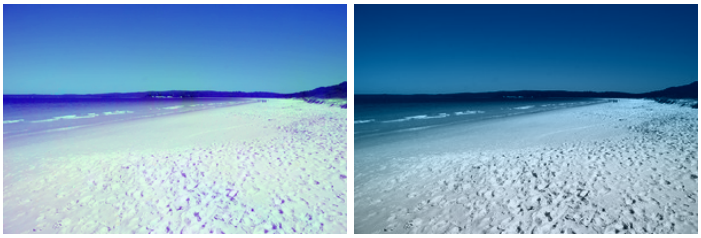
On compare notre modèle avec celui dont on s'inspire, celui de Bugeau *et al.* [1]. La première comparaison concerne simplement l'initialisation. On initialise l'algorithme avec la moyenne des candidats en RGB et la moyenne des candidats en YUV. On observe déjà une différence (Figure 18).

(a) Modèle RGB. (b) Modèle RGB projeté sur la contrainte de gris. (c) Modèle YUV (Bugeau *et al.*[1]), transformé dans l'espace RBG.

FIGURE 18 – Moyenne des candidats et projection sur la contrainte de gris. Dans le modèle RGB, les candidats extraits n'ont pas forcément la bonne luminance. Nous projetons donc sur la contrainte de luminance, fournie par l'image cible. Il n'y a pas de différence entre les deux images de droite.

La contrainte de luminosité n'est pas respectée lorsque l'on moyenne les candidats RGB, ce qui donne un résultat très peu cohérent, qui le devient lorsque l'on projette sur la contrainte de gris. À ce niveau là, les images sont quasiment identiques, ce qui montre que la recherche des candidats fournit les mêmes informations de teintes, et que l'algorithme de projection sur le niveau de gris fonctionne effectivement et avec stabilité. (Figure 18). On sait que le résultat n'est pas parfaitement identique car la projection en niveau de gris est orthogonale, mais le fait d'utiliser Y directement sans changer U et V n'est pas une projection orthogonale, car la matrice qui permet de passer de RGB à YUV n'est pas une matrice orthogonale.

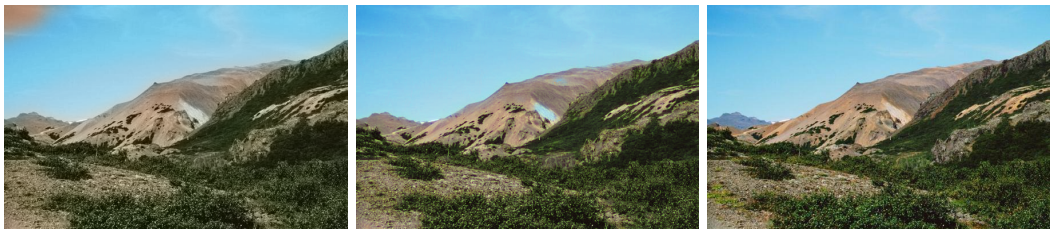
Une fois que l'algorithme a convergé, les couleurs initiales sont restées les mêmes. Dans le cas du modèle RGB, les teintes sont retrouvées de manière plus fidèle à la réalité (Figure 19).



(a) Modèle RGB. (b) Modèle YUV.

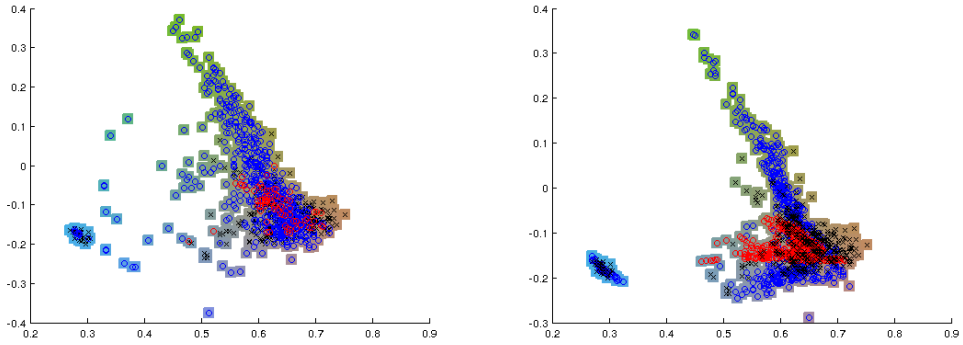
FIGURE 19 – Colorisation après convergence des algorithmes.

On arrive donc au constat que nous avons bien réglé le problème du caractère terne des images. Afin d'être sûr de nos résultats, on s'affranchit des erreurs causées par la recherche des candidats par le procédé suivant : on va coloriser une image en niveaux de gris en se servant pour image de référence (image source) la même image (auto-colorisation). Cela permettra de vérifier d'une part la stabilité visuelle de l'algorithme RGB, de comparer avec le modèle de Bugeau *et al.* [1] et enfin de comparer les nuages de points, Figure 20.



(a) Image re-colorisée par le modèle de Bugeau *et al.* [1]. (b) Image re-colorisée par le modèle RGB. (c) Image initiale.

FIGURE 20 – Comparaison des images colorisées par algorithme primal-dual sur le modèle de Bugeau *et al.* [1] et notre modèle au niveau de la répartition des couleurs sur le plan des luminances constantes.



(a) Nuages à l'initialisation.

(b) Nuages à convergence.

FIGURE 21 – Comparaison des nuages de points à l'initialisation et à convergence. Les axes représentent les coordonnées des points dans le repère arbitrairement choisis pour représenter le plan de luminance constante à plat.

3.3.4 Divers exemples d'images colorisées.

On a mis dans cette section des images naturelles colorisée par notre modèle. Nous présentons ici des images qui ont été bien colorisée et d'autre n'ayant pas été bien colorisée. Il s'agit en générale de la même scène ou du même objet photographié avec le même appareil photo numérique et identiquement redimensionné.

Pour commencer, des images bien colorisées sont visibles sur les figures 22 et 23. Les textures présentes sont clairement différentes, et le système de recherche de candidats les différencie bien.

FIGURE 22 – Ici, l'image est bien colorisée. Notons que l'on a ici augmenté la fréquence du sous-échantillonnage afin de récupérer les informations sur les petites structures comme les pâquerettes.

FIGURE 23 – Ici, la variation totale vectorisée est élevée à cause d'un contour très fractal.

Un problème récurent sur ce modèle est la confusion entre les zones d'arêtes et les textures. Mais les zones d'arêtes étant entre deux zones lisses, les candidats qui devraient

y être sélectionnés devrait l'être parmi les zones lisses et ce n'est pas le cas, ce qui amène des problèmes sur les zones d'arêtes, figures 24, 25 et 26.

(a) Image source. (b) Image cible. (c) Image colorisée.

FIGURE 24 – Des erreurs sont visibles au niveau des queues des feuilles. Ceci est dû au fait que les tiges sont différentes en terme de texture d'une partie lisse (le ciel) et donc sont plus proches des tuiles.

FIGURE 25 – Ici, un problème au niveau des branches, zones différentes des zones lisses mais qui sont spatialement proches.

FIGURE 26 – Le même problème de confusion contour/texture nous empêche ici de bien coloriser les fenêtres de la maison.

Le système de sélection des candidats est parfois faible et ne permet de distinction entre deux textures. Parfois, la distinction peut être faite sur une ou plusieurs métriques parmi les 8 retenues, mais en initialisant à la moyenne, on perd cette bonne information. C'est ce qui engendre les problèmes sur les figures 27, 28, 29, 30, 31, 32 et 33.

FIGURE 27 – Des problèmes sont visibles au niveau du jabot du paon et de sa queue. Les textures étant trop proches.

FIGURE 28 – Globalement, l'image est bien colorisée. En revanche, la texture de la laine du mouton et l'écorce des arbres sont parfois confondues avec l'herbe. Les poules n'ont pas été colorisées, ce qui peut être dû au sous-échantillonnage qui n'est pas assez dense.

FIGURE 29 – La texture de l'âne peut être visuellement confondue avec celle de l'herbe, et l'algorithme ne parvient pas à les discerner.

FIGURE 30 – Ici, la mauvaise colorisation du balai vient du fait que l'algorithme ne possède pas de système permettant de savoir si les textures sont orientées ou non.

FIGURE 31 – Les crêtes des poules sont mal colorisées à cause de leurs petites tailles, ce qui demanderai un sous-échantillonnage très fin, ce qui demande beaucoup de ressource de calcul.

FIGURE 32 – Exemple de confusion complète entre différentes textures qui ne sont pas lisses.

FIGURE 33 – Un peu de vert est visible au niveau de certaines zones (bois) à cause d'une autre confusion de texture.

Les parties lisses ne pourront jamais être différenciées, par aucun système. Et le nôtre échoue également sur les parties lisses comme on peut le voir sur les figures 34 et 35.

FIGURE 34 – Les parties lisses ne contiennent pas d'information texturale, et ne sont pas discernées par l'algorithme. C'est la raison pour laquelle la tôle sur laquelle dort Mimine, ne retrouve pas sa couleur d'origine. Notons que les images d'origine ont été redimensionnées à cause de la lenteur de l'algorithme. Notons que sur des images de définitions importantes (ici, elle provenait d'un appareil photo de résolution 3648×2736) les zones parfaitement lisses sont plus rares, et le sous-échantillonnage produit un effet de lissage qui fait perdre de l'information texturale.

FIGURE 35 – Les parties lisses ne sont pas différentiables, ce qui est ici bien illustré.

Enfin, notre système de recherche des candidats n'est pas robuste au bruit, et la variation totale vectorisée couplant les canaux, dans le cas de restauration d'images ancienne, la régularisation est peu robuste au bruit, comme cela est visible sur la figure 36.

FIGURE 36 – Ici, les problèmes proviennent d'une image cible très bruitée au niveau du ciel.

3.4 Proposition de critères numériques de qualité de colorisation et comparaison.

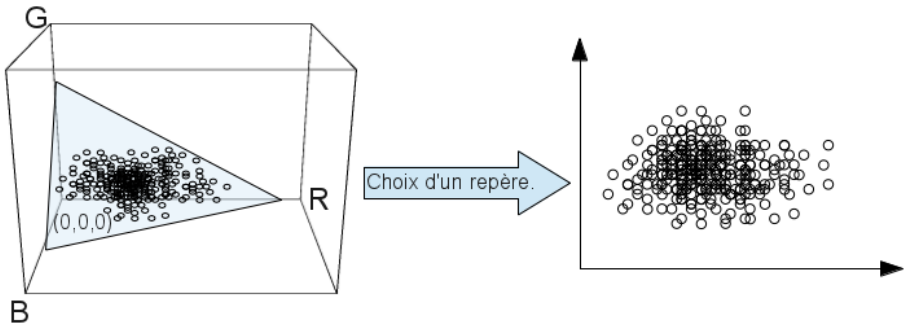


FIGURE 37 – Visualisation des points RGB sur un polygone du cube. Pour un niveau de luminance donné on trace les points RGB qui lui appartiennent en prenant trois points du polygone constitué de l'intersection du plan de luminance constante avec le cube RGB.

Il y a de grandes difficulté à obtenir un critère de qualité numérique fiable. Dans le cas de l'auto-colorisation, on possède une vérité terrain, et on peu donc calculer le PSNR, mais ce n'est pas un critère intéressant, car il ne quantifie qu'une différence, et pas de caractéristiques plus riches.

On peut, à partir de l'observation des nuages de points (définis dans la Section 2.5, voir figure 37), mise en relation avec la comparaison des images -et notamment leur côté terne-, déterminer des critères de qualité numérique de manière à pouvoir comparer objectivement la qualité de la colorisation et le côté terne des images. On veut, dans un premier temps savoir si les distributions des pixels sont plus ou moins proches de la distribution de l'image en niveaux de gris. On définit la distribution de l'image en niveaux de gris, pour un niveau de luminance donné, comme la distribution de la variable aléatoire constante égale à ce niveau de gris. Pour une image en couleur donnée on peut approximer par histogramme la distribution des couleurs appartenant à un niveau de luminance donné. On veut définir une distance entre ces deux distributions : celle d'une image en couleur et la distribution constante. Pour définir une telle distance on utilise la distance de transport [32] (The Earth Mover's distance) entre deux distributions.

Définissons quelques notions de transport optimal. Soit deux distributions de probabilités μ_0 et μ_1 définies comme suit :

$$\mu_0 = \sum_{i=1}^{n_0} p_i^0 \delta_{x_i^0} \tag{68}$$

et

$$\mu_1 = \sum_{i=1}^{n_1} p_i^1 \delta_{x_i^1} \ , \tag{69}$$

avec la distribution de μ_0 et μ_1 sont portées respectivement par $(x_i^0)_{i=1..n_0}$ et $(x_i^1)_{i=1..n_1}$, et $\sum_{i=1}^{n_1} p_i^1 = \sum_{i=1}^{n_0} p_i^0 = 1$, et $0 \leqslant p_i^k \leqslant 1$.

On définit la matrice C des coûts comme étant égale à :

$$C_{i,j} = ||x_i^0 - x_j^1||_2^2. \tag{70}$$

On définit P l'ensemble des couplages entre μ_0 et μ_1 :

$$P = \left\{ (\gamma_{i,j})_{i,j} \in \mathcal{M}_{n_0,n_1}(\mathbb{R}^+) \text{ tel que } \forall i, \sum_{j=1}^{n_0} \gamma_{i,j} = p_{0,i} \text{ et } \forall j, \sum_{i=1}^{n_1} \gamma_{i,j} = p_{1,j} \right\} , \quad (71)$$

et la distance de transport par :

$$\mathcal{W}(\mu_0, \mu_1) = \min_{\gamma \in P} \sum_{i,j} \gamma_{i,j} C_{i,j}. \quad (72)$$

Le γ réalisant ce minimum est noté γ^* et s'appelle le transport optimal entre μ_0 et μ_1.

On définit, avec ces notions, le critère **Écart-au-gris au niveau de luminance** x (que l'on abrègera *Écart-au-gris*) comme étant la distance de transport entre la distribution constante égale à celle du gris et la distribution des pixels de l'image ayant une luminance égale à x. Puisque l'on a un nuage de point, on approximera cette distribution par une variable aléatoire discrète ayant une distribution proche (principe de l'histogramme). Ce critère mesure la différence de répartition des pixels purement en terme de colorisation. Ce critère sera d'autant plus grand que la richesse des couleurs sera grande. Néanmoins ce n'est pas parfaitement représentatif de la qualité de la colorisation. Mais il est intéressant que, dans le cas de la colorisation d'une image par elle même (auto-colorisation), ce critère varie le moins possible par rapport à l'image d'origine.

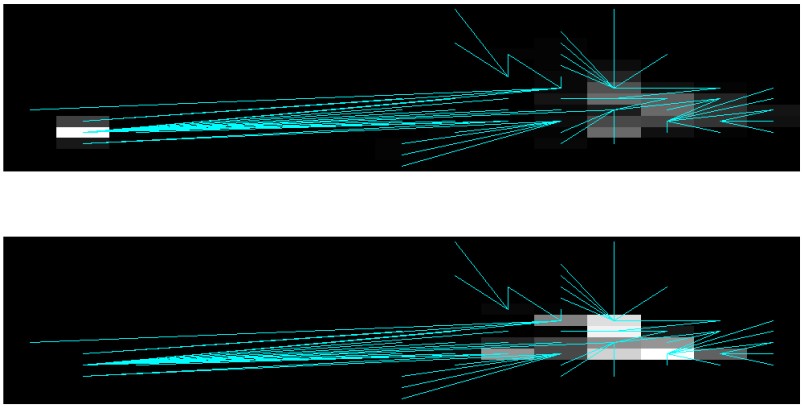


FIGURE 38 – Comparaison de deux distributions et flux de transport en cyan. Pour obtenir ces distributions on reprend les nuages de point de couleur à luminance constante, puis on calcule un histogramme en deux dimensions, normalisé (carrés en gris clair) puis on calcule le transport entre les deux distributions. En haut, la distribution de l'image initiale, en bas la distribution de l'image colorisée par Bugeau *et al.* [1]. En cyan, identique sur les deux images, le flux de transport entre les deux distributions.

Pour avoir une notion de stabilité dans le cas de l'auto-colorisation on peut définir un critère dérivé qui sera la distance de transport entre l'image de départ et l'image colorisée. On appellera ce critère **Distance-au-gris au niveau de luminance** x (que l'on abrègera *Distance-au-gris*). Comme pour l'écart au gris, ce calcul se fera sur une version discrétisée de la distribution, en l'occurrence sur une même grille de discrétisation. Notons que ce critère a un sens dans le cas de l'auto-colorisation, mais dans le cas d'un colorisation source-cible, une variété de texture et de couleur plus riche (et donc une inférence plus riche) au niveau de l'image source ne nous permettra pas d'appliquer convenablement notre critère (voir figure 38).

Toujours dans la cadre de l'auto-colorisation, il peut être intéressant de mesurer l'étalement de la distribution à convergence de l'algorithme : on peut définir l'**Étalement-au-gris au niveau de luminance** x, la distance de transport entre la distribution de la moyenne et la distribution elle-même. Ce qui est représentatif de son étalement. Notons que nous aurions aussi pu utiliser la variance, mais ce n'est pas très représentatif si les distributions ne sont pas unimodales.

Un autre critère plus simple basé sur les cartes de transport pourrait être définit comme la moyenne des minimums de la carte de transport, pris en ligne et en colonne. Cela donne un critère de rapprochement des distributions plus simple à calculer que la distance de transport, et donnant un critère assez similaire.

Tous ces critères vont permettre d'avoir des indicateurs numériques de la qualité de la colorisation. Nous avons travaillé sur l'algorithme au niveau de la régularisation de l'image, et nous avons vu que le plus important dans cette partie est la stabilité de la colorisation en terme de saturation : il faut éviter d'obtenir une image terne à convergence ou d'avoir une image saturée. La distance au gris par rapport à l'image originale doit être la plus petite possible afin de garantir une meilleure similarité, entre la distribution de l'image originale et l'image colorisée. Cela montrera la stabilité de l'algorithme. L'écart-au-gris permet de mesurer l'impression terne de l'image. Sa valeur doit être proche de celle de l'image originale, afin de s'assurer que l'algorithme ne produit pas des couleurs qui n'existent pas dans l'image originale mais également n'en élimine pas. L'étalement au gris permet moins de mesurer la stabilité de l'algorithme, mais plus de connaitre la richesse des couleurs des images produites de manière plus absolue. En calculant de tels critères, on obtiendra des indicateurs (voir tableau 3).

Modèle, image concernée	Écart-au-gris ($\times 10^{-2}$)	**Distance-au-gris par rapport à l'image originale ($\times 10^{-2}$)**	Étalement-au-gris ($\times 10^{-2}$)	**Minimum sur la carte des transports ($\times 10^{-2}$)**
Image originale	13.86	0	23.07	0
auto-colorisation par modèle de Bugeau *et al.* [1].	6.74	8.55	9.8	2.78
auto-colorisation par modèle RGB proposé.	18.86	5.36	28.80	0.8

TABLE 3 – Valeur des critères numériques pour un niveau de luminance de 150.

On remarque donc, si l'on considère la colonne de la Distance-au-gris entre l'image originale et les images obtenues par les modèles RGB et UV, que le modèle RGB est plus fidèle au niveau de la conservation de la distribution des couleurs. On remarque également que la moyenne des minima de la matrice de plan de transport est représentative, et va dans le même sens que la distance-au-gris.

Notons que pour un niveau de luminance donné, on peut avoir un nuage petit, *i.e.*, contenant peu de points, et on obtient donc un histogramme peu représentatif de la distribution, car l'approximation d'une distribution d'une variable aléatoire par l'histogramme d'un échantillon est asymptotique (Théorème de Glivenko-Cantelli, [33]). Ainsi on doit vérifier que la taille des échantillons est représentative pour calculer les distances de transport à fin de comparaison.

Afin de déterminer si une colorisation est meilleure qu'une autre, on peut se baser sur le critère **Distance-au-gris** : deux images ont une colorisation proche si les distributions de leurs couleurs sont proche. Ainsi, en colorisant une image, que l'on aura préalablement convertie en une image en niveaux de gris, et en se servant de l'image elle même comme source d'information, on peut comparer la distribution des couleurs. Il faut que l'image

Modèle, image concernée	Distance-au-gris par rapport à l'image originale ($\times 10^{-2}$)
Image originale	0
auto-colorisation par modèle de Bugeau *et al.* [1].	38.8
auto-colorisation par modèle RGB proposé.	14.0

TABLE 4 – Valeur moyenne calculée pour les 10 luminances conduisant aux plus grands nuages de point.

colorisée ait une distribution la plus proche possible de l'image initiale. On va procéder concrètement en calculant la distance-au-gris entre l'image initiale et l'image colorisée. Afin de comparer deux systèmes de colorisation, on dira que le système donnant l'image la plus proche de l'image initiale au sens de la Distance-au-gris est le plus stable. On remarque que si l'on compare pour tous les nuages de points, on arrive à environ 50% de nuages favorables au système RGB, ce qui ne nous permet pas de conclure. Comme mentionné dans le paragraphe précédent, on se servira des nuages contenant le plus grand nombre de points afin d'avoir un échantillon représentatif. On calcule donc la moyenne des distances-au-gris entre l'image initiale et l'image colorisée, pour les 10 plus grands nuages de points, pondérée par le cardinal de ces nuages. Cela nous donnera ainsi des valeurs chiffrées représentatives permettant de décider de la qualité de la colorisation. On peut voir le résultat en tableau 4. Les résultats obtenus nous permettent de dire que le modèle RGB colorise de manière plus fidèle que le modèle de Bugeau *et al.* en terme de distribution statistique des couleurs.

3.5 Quelques améliorations.

On donne dans cette section, quelques techniques efficaces pour améliorer visuellement les résultats. Il s'agit d'améliorations liées à la non-convexité du problème. Commençons avec une première section dédiée au problèmes liés à la non-convexité du problème.

3.5.1 Problèmes liés à la non-convexité.

Comme on l'a vu pour l'algorithme 12, on applique des méthodes qui convergent sous condition que les fonctionnelles soient convexes, à des problèmes qui ne le sont pas. On peut sur des exemples synthétiques voir les enjeux liés à ces problématiques naturelles :

On applique (algorithme 10) par exemple un algorithme de Forward-Backward (gradient projeté) au problème suivant :

$$\min_{(x,y)\in[0,1]^2} x(1-x)+y(1-y).$$

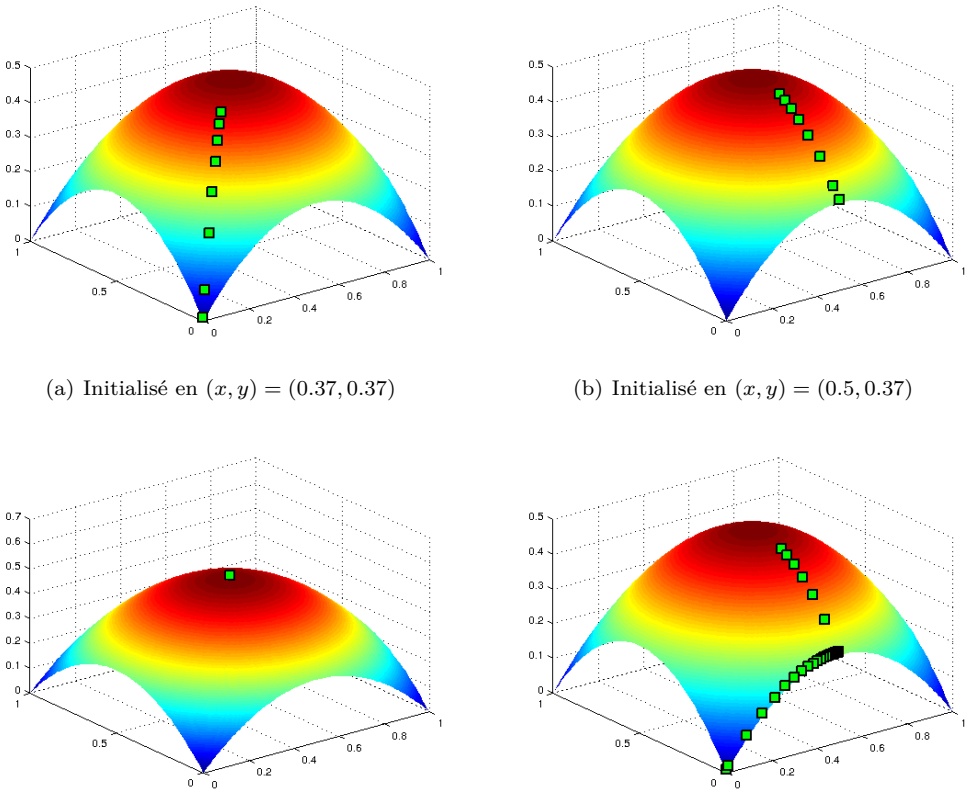
L'algorithme est donné en 10

Algorithme 10 Exemple synthétique.

1: $\gamma \leftarrow 0.1$
2: **for** $i = 1$:itérations **do**
3: $\quad x \leftarrow x + \gamma(2x-1)$
4: $\quad y \leftarrow y + \gamma(2y-1)$
5: $\quad (x,y) \leftarrow PA(x,y)$
6: **end for**

Où PA est la projection sur le carré $[0,1]^2$.

Les résultats diffèrent suivant les initialisations (voir figure 39).



(a) Initialisé en $(x, y) = (0.37, 0.37)$

(b) Initialisé en $(x, y) = (0.5, 0.37)$

(c) Initialisé en $(x, y) = (0.5, 0.5)$

(d) Initialisé en $(x, y) = (0.499, 0.36)$

FIGURE 39 – Gradient projeté pour minimiser le problème $\min_{(x,y)\in[0,1]^2} x(1-x)+y(1-y)$, qui est une version très simplifiée (dimension beaucoup plus petite) du problème que nous avons pour la colorisation.

Ces exemples nous montrent les choses suivantes : il ne suffit pas d'être un point fixe de l'algorithme pour être un minimum ou un minimiseur de la fonctionnelle. Il semblerait qu'il faille être un minimum attractif, *i.e.* un point tel que $\exists \varepsilon > 0$ tel que $\forall x_0 \in B(x_{min}, \varepsilon)$ la suite définie par $x_{n+1} = P(x_n - \nabla F(x_n))$ converge vers un unique point fixe x_{min}. Notons aussi que la fonctionnelle n'est pas convexe, ce qui nécessiterait de trouver une preuve de convergence du Forward-Backward dans des cas plus généraux (pseudo-convexité, quasi-convexité etc.)

Puisque le résultat de l'algorithme dépend fortement de l'initialisation, on modifie l'initialisation.

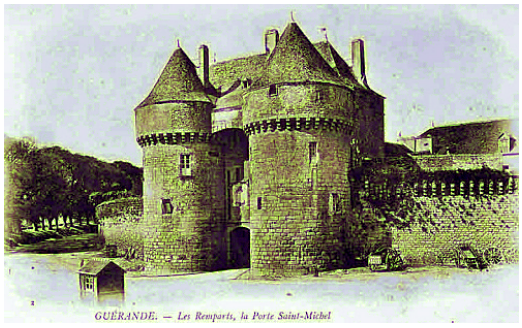
3.5.2 Le problème de l'initialisation.

Bugeau *et al.* [1] initialisent leur algorithme primal-dual à la moyenne des candidats. On se rend compte que c'est parfois un mauvais choix, en particulier lorsque 3 candidats sur les 8 sont mauvais et de grandes valeurs, la moyenne peut donner un résultat loin de la bonne valeur. Notre fonctionnelle étant non-convexe, le résultat est fortement dépendant de l'initialisation. Ce problème explique notamment l'apparition de la tâche en haut à gauche de l'image paysage (figure 21). Cela crée aussi quelquefois de fausses couleurs, c'est-à-dire des couleurs qui n'existent pas dans l'image de départ (source). On tente de remédier à cela en prenant la médiane, canal par canal. Une telle tâche est alors éliminée. En revanche cela pose toujours des problèmes de fausses couleurs. Ces fausses couleurs apparaissent dès l'initialisation et sont éliminées à convergence de l'algorithme. Ce phénomène advient aussi bien avec la moyenne que la médiane. En revanche, le fait d'avoir des fausses couleur dès l'initialisation est un peu inquiétant, car on peut craindre

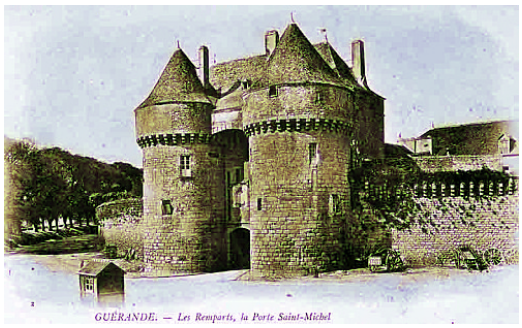
que l'algorithme converge vers un minimum local contenant des couleurs fausses.

Par exemple, on peut observer sur la figure 40 que la médiane et la moyenne donnent des couleurs trop jaunes par rapport à l'attache aux données beaucoup plus proche des vraies couleurs puisqu'elle est somme des candidats initiaux avec des coefficients quasiment tous égaux à 0 ou à 1. On remarque que l'algorithme semble naturellement corriger ces problèmes. On s'attachera à l'avenir, à réfléchir à des méthodes d'initialisation plus intelligentes.

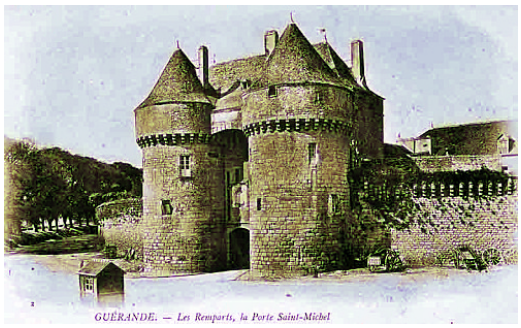
(a) Médiane des candidats.



(b) Moyenne des candidats.



(c) Image colorisée.



(d) Attache aux données : $\sum_i w_i C_i$.

FIGURE 40 – Comparaison des initialisations.

3.5.3 Pré-traitement des candidats.

Le modèle *RGB* permet la mise en place d'un pré-traitement des candidats, présenté dans la suite.

On dispose de 8 candidats par pixel. En revanche certains candidats ne sont pas d'un niveau de luminance correct. Ceci est dû au fait que l'on travaille en *RGB*, et que l'on extrait les candidats de l'image source sans discrimination du niveau de gris. Ce qui procure un avantage au niveau de l'invariance vis-à-vis du niveau de gris, mais qui ici amène un léger problème à résoudre. Pour ceux qui sont au-dessus de la valeur de luminance que l'on doit l'imposer, ce n'est pas grave, la projection sur le niveau de luminance imposé se fera sans changement de teinte.

REMARQUE *:* 3.5. Il faut noter que l'une des trois lois de Grassmann sur la colorisation dit que deux teintes sont les mêmes si les vecteurs des trois primaires sont proportionnels [7].

En revanche dans le cas où le niveau de gris à imposer est au-dessus de celui du candidat, il pourra parfois être impossible d'imposer un niveau de luminance sans changer la teinte.

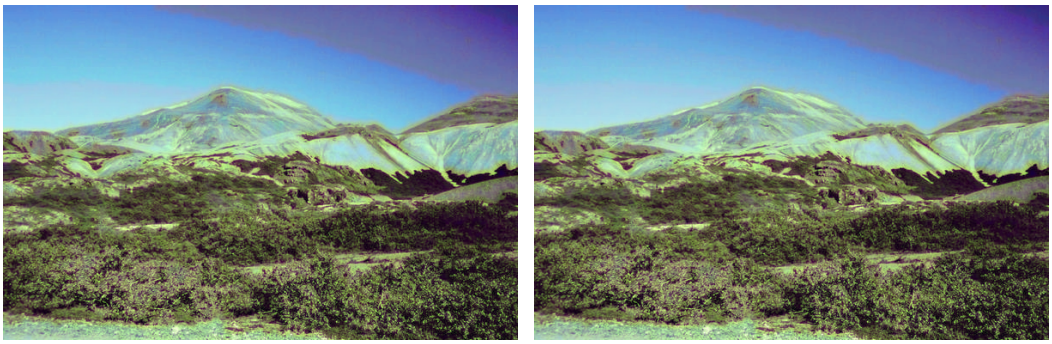
Exemple 3.6. *Si le candidat est* $(255, 0, 0)$ *(du rouge pur), et que le niveau de gris est* 255 *on ne peut pas trouver un vecteur RGB qui soit à la fois proportionnel à ce vecteur, de luminance* 255 *et contenu dans le cube* $[0, 255]^3$.

7. `http://fr.wikipedia.org/wiki/Lois_de_Grassmann`

On décide d'effectuer le pré-traitement suivant : s'il existe de tels candidats, on met à 0 le poids w leur correspondant, puis on re-projette W sur le simplexe. En agissant ainsi, on n'empêche pas l'algorithme d'utiliser ce candidat, mais on le lui déconseille vivement. Les résultats obtenus par cette méthode sont bien améliorés (figures 41 et 42). Il n'y a plus de bleu sur le sable, et sur les montagnes de droite. Ces couleurs étaient aberrantes, par exemple le sable ne peut être bleu. En les éliminant, on rend l'image plus réaliste.

(a) Sans préprocessing. (b) Avec préprocessing.

FIGURE 41 – Résultat de l'algorithme primal-dual sur l'image Beach, avec et sans pré-traitement.



(a) Sans préprocessing. (b) Avec préprocessing.

FIGURE 42 – Résultat de l'algorithme primal-dual sur l'image Paysage, avec et sans pré-traitement.

3.6 Critiques des méthodes et justification d'un paradoxe du modèle.

On peut émettre quelques critiques sur le modèle et l'algorithme. Une première critique facile est la faiblesse de l'algorithme sur les zones lisses. Ce qui est parfaitement normal. En effet, sur les parties lisses, il n'y a pas de caractéristique texturale permettant de faire de lien entre l'image en niveaux de gris et des teintes de couleur. Ce problème n'est pas lié au modèle posé, mais à un problème inhérent à la colorisation basée-exemple. Nous n'avons pas proposé pour l'instant de méthode permettant à l'utilisateur d'inférer avec le système et la fonctionnelle, telle qu'elle est posée ne se prête pas facilement à ce jeu. On pourrait, sans beaucoup de modifications, proposer une restriction de la zone de recherche comme cela est proposé par Welsh *et al.* [10]. Ceci sera envisagé dans le futur. Diverses façons d'aborder le problème pourraient être envisagées. Notamment une méthode de scribles posées par l'utilisateur, suivi d'une initialisation de certains candidats tel que cela peut être fait par Sapiro [9]. On peut envisager une solution de la manière suivante : l'utilisateur pose des points colorés. Ensuite on définit la distance entre deux

points suivant les irrégularités de l'image. Par exemple [9] $\min_{C_b} \int_\Omega \rho(||\nabla Y - \nabla B_b||)d\Omega$, avec ρ une norme-2, ou une norme-1. Ce qui fait que deux points d'une même partie lisse sont considérés comme étant à distance nulle l'un de l'autre, et doivent donc être de même couleur. Cela peut donc permettre de modifier les candidats c_i en les définissant par cette méthode. Cela permettrait ainsi à l'utilisateur de coloriser les parties lisses sans rendre le modèle trop compliqué.

Notons également que ce modèle présente un paradoxe non-négligeable : en effet, cette méthode de recherche des candidats fonctionne bien sur les images texturées pour obtenir de l'information à partir de l'image en niveaux de gris, afin de déterminer les couleurs qu'il faut mettre. En revanche, la variation totale a tendance à lisser les images, même si elle permet la préservation des contours. Il y a donc un paradoxe. On utilise un modèle peu adapté aux textures sur des images texturées car l'inférence nous y contraint. Néanmoins les résultats sont d'une allure visuelle correcte. On peut l'expliquer : d'une part, la contrainte de gris (luminance) étant maintenue, les textures qui apparaissent dans le canal de luminance sont maintenues. Or l'œil est plus sensible à ce canal. Donc, le maintient strict de cette contrainte permet de dissimuler les problèmes dûs au modèle basé sur la minimisation de la variation totale. Ensuite, certains attributs texturaux sont calculés sur des patchs de petite taille (la variance). Ce qui fait que les textures qui sont peu fines au regard de la taille des patchs seront préservées. Finalement, le modèle ne semble pas souffrir de ce paradoxe sur les exemples utilisés. Mais il existe peut-être des images d'une telle texture qui ne soit pas bien modélisées par ce modèle, et pour lesquelles on obtient des résultats incohérents.

4 Conclusion.

Dans ce rapport nous avons écrit un modèle et donné un algorithme qui calcule la solution de la fonctionnelle proposée. La difficulté principale a été, dans un premier temps, d'implémenter la projection sur la contrainte $\chi_{[0,255]} + \chi_{Y(u)=I_g}$. Des essais alternatifs évitant ce calcul ont été testés, mais ils ont tous été abandonnés à cause d'un manque de résultat. Bien que ces méthodes minimisaient la fonctionnelle, elles n'étaient pas assez rapides. La fonctionnelle n'étant pas convexe, il n'y a pas forcément unicité du minimum et garantie de convergence de l'algorithme vers un minimum local. Nous ne savons donc pas si un minimum local est atteint après convergence. De plus, le problème pratique essentiel est que l'image à convergence n'est pas forcément celle qui est la plus pertinente en terme de qualité visuelle.

L'implémentation de l'algorithme dans l'espace de couleur RGB a apporté une amélioration substantielle. Puis en modifiant la fonctionnelle au niveau du terme de régularisation, en posant le terme de variation totale vectorisée, nous avons encore affiné la qualité des résultats, notamment en permettant une amélioration des contours. Ces derniers étaient dégradés au niveau des zones de changement de direction brutale. Les quelques améliorations apportées au niveau de l'initialisation ont également permis d'obtenir des résultats de meilleure qualité.

Les images colorisées par le modèle de Bugeau at al. [1] étaient trop ternes. Nous avons apporté une explication à cela, et nous avons posé un critère numérique permettant de quantifier l'amélioration dûe au modèle.

Les travaux qui restent à mener sont multiples. Dans un premier temps, il serait intéressant de diminuer le temps de calcul. Pour cela, il serait utile de coupler les trois canaux en YUV, pour des raisons évidentes de coût en temps de calcul. Ensuite on pourrait améliorer les résultats de la minimisation de fonctionnelles non-convexes, notamment par l'adjonction d'un terme aléatoire. Enfin, il faudrait penser à ajouter une possibilité pour l'utilisateur d'interagir avec le système, par exemple par ajout manuel de points colorés sur l'image comme dans l'article de Sapiro [9].

Références

[1] A. Bugeau, V.-T. Ta, N. Papadakis *et al.*, "Variational exemplar-based image colorization," 2013.

[2] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," in *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3. ACM, 2004, pp. 689–694.

[3] W. Markle and B. Hunt, "Coloring a black and white signal using motion detection," Jul. 5 1988, uS Patent 4,755,870.

[4] R. Irony, D. Cohen-Or, and D. Lischinski, "Colorization by example," in *Proceedings of the Sixteenth Eurographics conference on Rendering Techniques*. Eurographics Association, 2005, pp. 201–210.

[5] D. Nie, Q. Ma, L. Ma, and S. Xiao, "Optimization based grayscale image colorization," *Pattern Recognition Letters*, vol. 28, no. 12, pp. 1445–1451, 2007.

[6] T. Horiuchi, "Colorization algorithm using probabilistic relaxation," *Image and Vision Computing*, vol. 22, no. 3, pp. 197–202, 2004.

[7] T. Takahama, T. Horiuchi, and H. Kotera, "Improvement on colorization accuracy by partitioning algorithm in cielab color space," in *Advances in Multimedia Information Processing-PCM 2004*. Springer, 2005, pp. 794–801.

[8] L. Yatziv and G. Sapiro, "Fast image and video colorization using chrominance blending," *Image Processing, IEEE Transactions on*, vol. 15, no. 5, pp. 1120–1129, 2006.

[9] G. Sapiro, "Inpainting the colors," in *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, vol. 2. IEEE, 2005, pp. II–698.

[10] T. Welsh, M. Ashikhmin, and K. Mueller, "Transferring color to greyscale images," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 277–280, 2002.

[11] G. Di Blasi and D. Reforgiato, "Fast colorization of gray images," *Eurographics Italian*, 2003.

[12] G. Charpiat, M. Hofmann, and B. Schölkopf, "Automatic image colorization via multimodal predictions," in *Computer Vision–ECCV 2008*. Springer, 2008, pp. 126–139.

[13] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf : Speeded up robust features," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 404–417.

[14] T. Chen, Y. Wang, V. Schillings, and C. Meinel, "Grayscale image matting and colorization," in *Proceedings of Asian Conference on Computer Vision (ACCV)*. Citeseer, 2004.

[15] D. L. Ruderman, T. W. Cronin, and C.-C. Chiao, "Statistics of cone responses to natural images : Implications for visual coding," *JOSA A*, vol. 15, no. 8, pp. 2036–2045, 1998.

[16] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. Springer, 2011, pp. 185–212.

[17] P. L. Combettes and V. R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling & Simulation*, vol. 4, no. 4, pp. 1168–1200, 2005.

[18] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex analysis and minimization algorithms*. Springer, 1993.

[19] P. L. Combettes and J.-C. Pesquet, "A douglas–rachford splitting approach to nonsmooth convex variational signal recovery," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, no. 4, pp. 564–574, 2007.

[20] H. Raguet, J. Fadili, and G. Peyré, "Generalized forward-backward splitting," *arXiv preprint arXiv :1108.4404*, 2011.

[21] A. Chambolle, "An algorithm for total variation minimization and applications," *Journal of Mathematical imaging and vision*, vol. 20, no. 1-2, pp. 89–97, 2004.

[22] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of Mathematical Imaging and Vision*, vol. 40, no. 1, pp. 120–145, 2011.

[23] F. Pierre, *Études théorique de quelques fonctionnelles de EDP utilisée en traitement d'images*. Rapport de projet, 2013.

[24] L. Ambrosio, N. Fusco, and D. Pallara, *Functions of bounded variation and free discontinuity problems*. Clarendon Press Oxford, 2000.

[25] B. Goldluecke and D. Cremers, "An approach to vectorial total variation based on geometric measure theory," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 327–333.

[26] A. Kolmogorov and S. Fomine, "Éléments de la théorie des fonctions et de l'anaylse fonctionelle (2nd edn.) mir," 1977.

[27] J. Fresnel, *Espaces quadratiques, euclidiens, hermitiens*. Hermann, 1999.

[28] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM review*, vol. 52, no. 3, pp. 471–501, 2010.

[29] V. Retakh, "A theorem of eidelheit," *Siberian Mathematical Journal*, vol. 17, no. 6, pp. 999–1009, 1976. [Online]. Available : http://dx.doi.org/10.1007/BF00968028

[30] Y. Nievergelt, "Schmidt-mirsky matrix approximation with linearly constrained singular values," *Linear algebra and its applications*, vol. 261, no. 1, pp. 207–219, 1997.

[31] Y. Chen and X. Ye, "Projection onto a simplex," *arXiv preprint arXiv :1101.6081*, 2011.

[32] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.

[33] D. Dacunha-Castelle and M. Duflo, "Probabilités et statistiques, tome 2, problèmes à temps mobile," 1983.

[34] F. S. Hillier and G. J. Lieberman, *Introduction to Mathematical Programming*. McGraw-Hill, 1990.

[35] S. Anthoine, J.-F. Aujol, Y. Boursier, and C. Mélot, "Some proximal methods for poisson intensity cbct and pet," *Inverse Problems and Imaging*, vol. 6, no. 4, pp. 565–598, 2012.

A Méthodes alternatives testées.

A.1 Résolution par Forward-Backward généralisé.

Le modèle (54) a l'inconvénient d'être non-convexe, de posséder des minimas locaux, des point-selles et des maxima locaux. Ceci est dû au terme $\frac{\alpha}{2}\int_\Omega \sum_{i=1}^{8} w_i(1-w_i)$ qui n'est pas convexe. Néanmoins, la fonctionnelle $F(U,W)$ est convexe en u, donc si l'on veut minimiser cette fonctionnelle en u, il existe des méthodes dans la littérature permettant de le faire. Minimiser la fonctionnelle $F(u,W)$ par rapport à u revient à minimiser :

$$\tilde{F}(u) := TV(u) + \frac{\lambda}{2}\int_\Omega \sum_{i=1}^{8} w_i||u-c_i||^2 + \chi_{u\in[0,255]} + \chi_{Y(u)=I_g}. \tag{73}$$

On distingue dans cette fonctionnelle deux types de termes : un terme lisse, différentiable : $\frac{\lambda}{2}\int_\Omega \sum_{i=1}^{8} w_i||u-c_i||^2$ et un terme non lisse, mais convexe : $TV(u) + \chi_{u\in[0,255]} + \chi_{Y(u)=I_g}$. Raguet *et al.* [20] fournit un algorithme convergent pour minimiser ce type de fonctionnelle : il s'agit du Forward-Backward généralisé. On le met donc en application dans le cas de cette fonctionnelle pour minimiser F par rapport à u. Cela donne l'algorithme 11.

Algorithme 11 Forward-Backward généralisé, appliqué à la colorisation.

$p_1, p_2, p_3 \leftarrow \frac{1}{3}$.
γ, λ_t calculés automatiquement.
$z_1, z_2, z_3 \leftarrow u$
$X \leftarrow p_1 z_1 + p_2 z_2 + p_3 z_3$
for i=1 :itérations **do**
  $z_1 \leftarrow z_1 + \lambda_t \left(PA(2 - z_1 - \gamma\lambda(2X - 2X - 2\sum_i w_i c_i)) - X\right)$
  $z_2 \leftarrow z_2 + \lambda_t \left(proxTV_{\frac{\gamma}{p_2}}(2X - z_2 - \gamma\lambda(2X - 2\sum_i w_i c_i)) - X\right)$
  $z_3 \leftarrow z_3 + \lambda_t \left(PG(2X - z_3 - \gamma\lambda(2X - 2\sum_i w_i c_i)) - X\right)$
  $X \leftarrow p_1 z_1 + p_2 z_2 + p_3 z_3$
end for

$proxTV_\lambda$ désigne la fonction qui à $\tilde{u}$ associe l'unique minimum de la fonctionnelle en u :

$$\frac{||u-\tilde{u}||^2}{2\lambda} + TV(u). \tag{74}$$

Cet algorithme converge vers le minimum à W fixé. Notons que le choix des poids p_i est laissé libre à l'utilisateur, entre 0 et 1, strictement, et dont la somme vaut 1. Le choix de ces paramètres n'influe pas sur la solution, mais il influe sur la vitesse de convergence.

Il faut aussi minimiser par rapport à la variable W, à u fixé, ce qui revient à minimiser la fonctionnelle :

$$F(W) := \frac{\lambda}{2}\int_\Omega \sum_{i=1}^{8} w_i||u-c_i||^2 + \frac{\alpha}{2}\int_\Omega \sum_{i=1}^{8} w_i(1-w_i) + \chi_{W\in\Delta}\ , \tag{75}$$

qui est composée d'un terme différentiable et d'un terme convexe non différentiable, qui correspond à une projection sur un simplexe. En revanche la fonctionnelle étant non-convexe, on n'a pas d'algorithme permettant de calculer un minimum de la fonctionnelle, ni même d'algorithme convergent vers un minimum. Il faut néanmoins envisager d'appliquer un algorithme issu de l'analyse convexe, l'algorithme de gradient projeté (Algorithme 12). Dans cet algorithme, P_Δ désigne la projection sur le simplexe

$\{(w_1, \ldots, w_n)$ tel que $0 \leqslant w_i \leqslant 1$, $\forall i \in [1..n]$ et $\sum_i w_i = 1\}$, fournie par Chen *et al.* [31].

Algorithme 12 Gradient projeté.

1: **for** i=1 :iterations **do**
2: $\quad W \leftarrow P_\Delta \left(W - \tau_w((||u - c_i||^2)_i + \alpha(1 - 2W)\right)$
3: **end for**

Pour l'instant on initialise l'algorithme à $W \leftarrow \frac{1}{8}$, mais on verra par la suite (figure 39) que ce choix peut parfois être le pire.

Maintenant que l'on dispose de ces deux algorithmes, on peut essayer de minimiser la fonctionnelle, en alternant une minimisation par rapport à u avec une minimisation par rapport à W, malgré le fait que l'on n'a aucune preuve de convergence. Un premier essai est appliqué sur les exemples beach et paysage (figure 43).

(a) Image beach. (b) Image paysage.

FIGURE 43 – Résultat de la minimisation de la fonctionnelle par l'algorithme de Forward-Backward généralisé [20].

Cet exemple est colorisé et l'aspect global est correct à première vue, mais l'on peut constater en regardant attentivement l'image que le sable contient beaucoup trop de rose. De plus on remarque (sur la ligne d'horizon) des petits artefacts. Il s'agit de pixels qui deviennent d'une couleur à dominante de teinte verte. Cette couleur n'était pas présente parmi les candidats initiaux, ce qui est donc une erreur artificielle (artefact).

Sur l'image paysage, les artefacts deviennent plus flagrants. On trouve du bleu très saturé sur les montagnes, ce qui donne un aspect peu réaliste à l'image colorisée.

A.2 primal-dual par projection approchée

Afin de mettre en place l'algorithme primal-dual et faire l'économie de la projection sur le polygone, j'ai tenté une approche itérative de la projection. On donne donc l'algorithme 13. En voyant cet algorithme, on peut se demander s'il existe de nombreux cas dans lesquels la première projection sort effectivement du cube RGB. Cela va fortement dépendre de la valeur de la luminance. On peut donner un exemple numérique dans lequel on aura un fort taux de sortie du cube RGB : par exemple, on pose $I_g = 10$. En testant tout les X possibles du cube RGB, on observe que la projection de X sur le plan $A^t X = I_g$ est à l'extérieur du cube dans 97.04% des cas.

Cet algorithme ne calcule pas la projection exacte. Il n'est capable que de s'en approcher. On converge vers cette solution en prenant un grand nombre d'itérations. Le maximum de l'erreur en fonction du nombre d'itérations de l'algorithme de projection par itération est montré en figure 44.

Algorithme 13 Projections itératives.

X donné.
for i=1 :itérations **do**
    $X \leftarrow \frac{A}{||A||^2}(I_g - \langle X|A\rangle) + X$
    if $X \geqslant 255$ **then**
        $X \leftarrow 255$
    else if $X \leqslant 0$ **then**
        $X \leftarrow 0$
    end if
end for

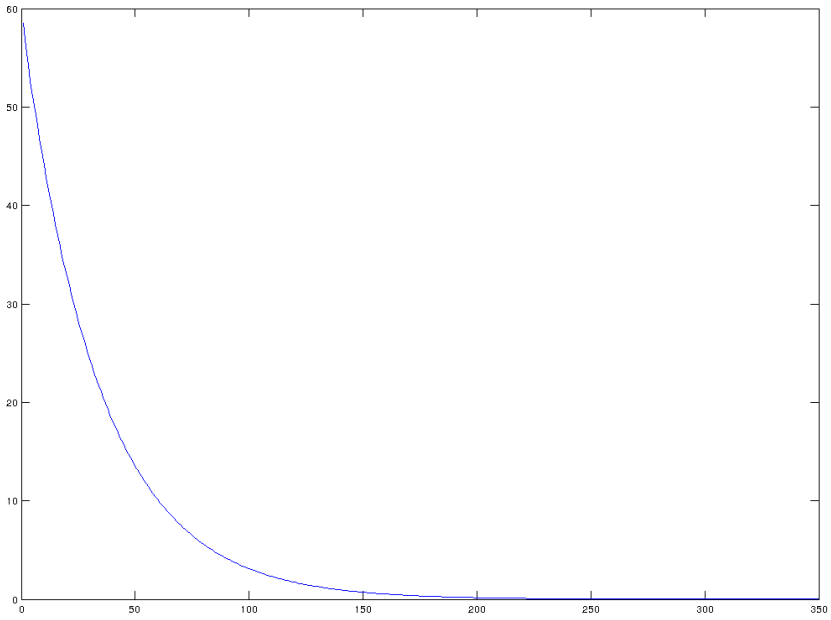


FIGURE 44 – Erreur commise (en dynamique = $\max(P_1 - P2) - \min(P_1 - P_2)$) par la projection itérative en fonction du nombre d'itérations. On compare ici par rapport à la projection exacte donnée dans la section 3.2.

Néanmoins, cet algorithme est lent si l'on veut une solution assez proche de la bonne solution.

Si l'on se contente d'obtenir une solution approchée pour travailler avec l'algorithme primal-dual le résultat est mauvais. En effet, si l'on se contente de 100 itérations, afin de conserver un temps de calcul semblable à la projection exacte, il faut se contenter d'une erreur maximale de 23% de la dynamique de l'image, ce qui n'est pas envisageable. Si l'on accepte de descendre à une erreur inférieure à $\frac{1}{255}$, (on réalise des tests avec une erreur maximale de 0.31% de la dynamique de l'image) on obtient un résultat certes meilleur, mais dont la qualité est encore trop éloignée de l'algorithme implémenté avec la projection exacte (figure 45), et surtout avec un temps de calcul trop long.

(a) Image Beach. (b) Image Paysage.

FIGURE 45 – Résultat donné par l'algorithme primal-dual lorsque l'on approxime la projection avec la projection itérative.

Cet algorithme donne un résultat équivalent à celui du Forward-Backward généralisé en terme de qualité, c'est-à-dire un résultat contenant des artefacts.

A.3 Un deuxième essais de Generalized Forward-Backward (GFB).

On a vu dans la section précédente que l'on pouvait avoir une projection directe sur la contrainte $\chi_{u\in[0,255]} + \chi_{Y(u)=I_g}$. On peut penser à appliquer le GFB en condensant les deux projections séparées en une seule, comme cela a été implémenté pour le primal-dual. L'algorithme devient alors le suivant :

Algorithme 14 Forward-Backward généralisé, appliqué à la colorisation, minimisant (73).

$p_1, p_2 \leftarrow \frac{1}{2}$.
γ, λ_t calculés automatiquement.
$z_1, z_2 \leftarrow u$
$X \leftarrow p_1 z_1 + p_2 z_2$
for i=1 :iterations **do**
    $z_1 \leftarrow z_1 + \lambda_t \left(PG(2 - z_1 - \gamma\lambda(2X - 2X - 2\sum_i w_i c_i)) - X\right)$
    $z_2 \leftarrow z_2 + \lambda_t \left(prox TV_{\frac{\gamma}{p_2}} (2X - z_2 - \gamma\lambda(2X - 2\sum_i w_i c_i)) - X \right)$
    $X \leftarrow p_1 z_1 + p_2 z_2$
end for

où $proxTV$ est défini en (74).

(a) Image Beach. (b) Image Paysage.

FIGURE 46 – Résultat de la minimisation de la fonctionnelle, donné par l'algorithme de Forward-Backward généralisé [20]. La contrainte de gris est calculée ici de manière exacte.

On remarque (figure 46) que les résultats comportent autant d'artefacts que la version précédente du Forward-Backward généralisé. Le problème peut venir du fait suivant : le FBG travaille sur deux canaux en parallèle qu'il moyenne ensuite (notons que cet algorithme peut être naturellement parallélisé), donc si le premier canal respecte bien la contrainte, il n'en est pas forcément de même pour le deuxième. On avait remarqué que si la projection n'était pas parfaite dans le cas du primal-dual, il survenait quelques artefacts : on a ici la même chose. La projection n'est pas parfaite et fait apparaitre des artefacts.