

Codage informatique des couleurs

Le codage d'un pixel peut se faire sur 32 bits, dont 24 bits sont utilisés pour coder la couleur, les 8 bits restants étant :

- soit inutilisés ;
- soit, avec les représentations (OpenGL, DirectX) et/ou les formats d'image qui le permettent (comme le PNG), à coder une information de transparence dite *alpha channel* (voir le standard RGBA). À travers ce pixel de l'image "passera" en partie la couleur d'un pixel d'une autre image placée dans la même fenêtre, mais « derrière » la première image (technique dite *alpha blending* en anglais).

Il est également possible d'utiliser une palette de 256 couleurs, auquel cas la couleur du pixel est codée sur 8 bits.

Sur les vieux écrans qui n'avaient que 8 bits de couleur par pixel, une palette était utilisée. Cette palette n'était pas universelle, chaque constructeur étant libre de définir cette palette à sa guise. Il existait toutefois un certain consensus lié à l'existence d'une sorte de standard HTML qui veut qu'une certaine palette dite "*palette web*" (en anglais "safe-web palette" ou "safety palette") devait être privilégiée : dans les faits, la quasi-totalité des navigateurs internet respectait cette palette, qui comporte 216 couleurs dont les trois composantes RVB (voir ci-dessous) sont l'un des 6 multiples de 51 suivants : 0, 51, 102, 153, 204 ou 255. Remarquez que :

$$216 = 6 \times 6 \times 6 = 6^3$$

Détails


Dans la première partie de l'article, nous ne nous intéresserons qu'aux **24 bits** de codage des couleurs. Les explications données correspondront donc non seulement à la représentation des couleurs sur 32 bits mais aussi à celle sur 24 bits.

Les 24 bits d'une couleur se décomposent en 3 fois 8 bits :

- 8 bits sont consacrés à la teinte primaire **rouge** ;
- 8 bits sont consacrés à la teinte primaire **vert** ;
- 8 bits sont consacrés à la teinte primaire **bleu**.

Une séquence de 8 bits permet de coder un nombre entier compris entre 0 et $V_{max} = 255$: en effet, 2^8 vaut 256. Par conséquent, la valeur de la composante rouge d'un pixel peut être représentée selon 256 niveaux différents (allant du 0, absence de rouge, à 255, rouge d'intensité maximum). Et il en est de même pour les 2 autres composantes primaires, le vert et le bleu.

Donnons un exemple :

Le carré ci-contre  est formé de pixels d'une couleur uniforme dont les caractéristiques RVB sont les suivantes :

- composante rouge : 251, soit en codage binaire (sur 8 bits) **11111011** ;
- composante verte : 208, soit **11010000** ;
- composante bleue : 151, soit **10010111**.

Le codage binaire sur 24 bits de cette couleur est donc le suivant :

111110111101000010010111.

Or il existe deux grandes familles de représentation des couleurs, telles qu'elles peuvent apparaître dans une image présentée sur un écran d'ordinateur : le codage RVB (ou RGB en anglais), dont les principes viennent d'être décrits, et le codage TSL (ou HSL en anglais).

Voyons donc à présent quelles sont les valeurs des 3 composantes du codage TSL de la couleur choisie précédemment, exprimées (comme c'est assez souvent le cas) selon une échelle allant de 0 à 240 :

- composante **Teinte** : 23 ;
- composante **Saturation** : 222 ;
- composante **Luminance** : 189.

Ici, on se rapportera utilement aux explications fournies dans l'article Teinte saturation lumière, dans lequel d'autres valeurs maximales sont choisies pour la teinte (de 0 à 360 °), la saturation (de 0 à 100 %) et la luminance (de 0 à 100 %).

Le codage RVB est celui qui est mis en œuvre dans de nombreux périphériques numériques : en entrée (scanner-couleurs, appareil photo numérique, caméscope...) comme en sortie (écran en couleurs, imprimante, quadrichromie, photocopieuse-couleurs...).

Le codage TSL, destiné aux opérateurs humains, est adapté à la caractéristique de leurs rétines : une personne entraînée peut d'ailleurs donner avec une approximation satisfaisante les valeurs TSL d'une couleur qu'on lui présente (la plus difficile à retrouver étant réputée être le marron); elle peut aussi, même si elle est novice, trouver assez rapidement, en s'aidant par exemple des outils de *Sélection de couleur* offerts dans de nombreux logiciels de dessin ou de retouche, les composantes TSL d'une couleur qu'elle n'a pas sous les yeux mais qu'elle *imagine* ; enfin, le *langage* TSL de définition des couleurs permet de définir facilement certains des dégradés de teinte que la représentation RVB ne permet pas de définir aussi facilement. En revanche, la très grande majorité des langages de développement exige d'utiliser le codage RVB pour la définition de la couleur d'un tracé, d'un fond de fenêtre, d'un texte, etc. ; c'est pourquoi il peut être utile de disposer de moyens permettant de passer d'un codage TSL à un codage RVB, et réciproquement.

La notion de couleur est parfois généralisée en ajoutant un indice de transparence appelé **Alpha**. Voir RGBA.

Avant de présenter ces moyens, nous allons d'abord présenter et commenter un outil de sélection de couleurs, ainsi que des exemples de dégradé entre 2 teintes.

Outils de sélection de couleur

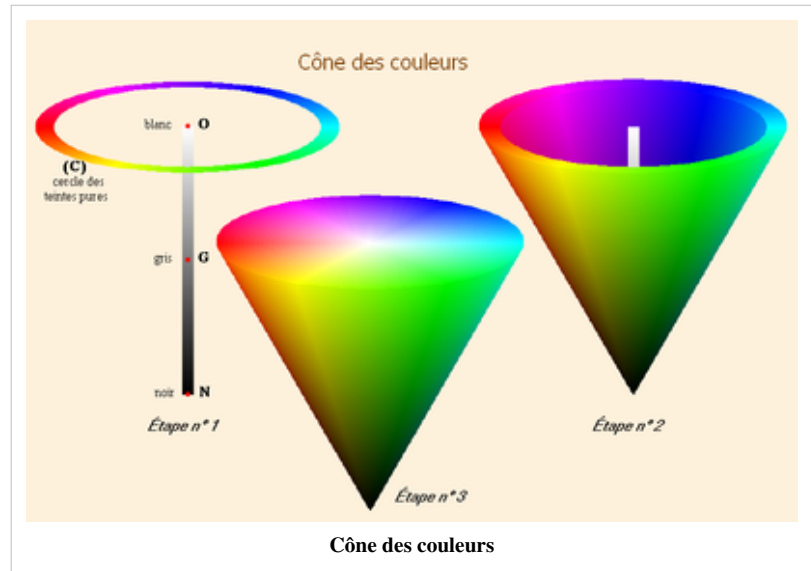
Un outil de sélection de couleur comporte en général au minimum 4 parties :

- 2 parties visuelles de choix dont l'une est un carré et l'autre un rectangle étroit dressé sur son petit côté,
- 1 partie visuelle d'affichage (petit rectangle rempli de la couleur choisie), et enfin
- 1 partie purement numérique donnant à la fois les composantes TSL et RVB de la couleur choisie par l'opérateur.

Voici (à gauche) comment se présente l'outil généralement présent dans les applications Microsoft, et (à droite), à titre de comparaison, l'outil équivalent du logiciel PhotoShop d'Adobe.

Pour comprendre comment fonctionne un sélecteur de couleurs, il est commode de se représenter l'ensemble des couleurs disponibles sous la forme suivante :

- imaginons un axe vertical sur lequel sont placés un point N de couleur noire, un point O de couleur blanche ;
- entre ces 2 points extrêmes, les points intermédiaires seront coloriés dans une teinte grise de la luminance intermédiaire qui convient (échelle régulière linéaire) : par exemple, le point G de la figure correspond au gris moyen (de luminance $L = \frac{L_{max}}{2}$) ;



- dans le plan horizontal passant par O, on trace ensuite un cercle (C) portant toute la gamme des teintes pures, c'est-à-dire de luminance maximale L_{max} et de saturation maximale S_{max} .
- Toutes les autres couleurs disponibles sur les écrans d'ordinateur sont intermédiaires entre les teintes qui viennent d'être décrites (noir, blanc, gamme des gris et gamme des teintes pures). Elles seront donc toutes situées à l'intérieur du cône d'axe NO qui passe par (C), ce qui correspond au coloriage en trois étapes représenté sur la figure suivante (cliquer sur cette figure pour avoir des explications géométriques supplémentaires) :

Le choix d'une couleur consiste donc à définir un point situé à l'intérieur de (ou sur) ce *cône des couleurs*.

Dans tous les cas, la définition d'une couleur par ses composantes TSL exige un triple choix qui doit nécessairement être réalisé par l'opérateur en 2 temps :

- Dans le cas du premier sélecteur de couleurs : l'opérateur doit savoir que le carré de choix représente le choix des 2 composantes T et S alors que le rectangle de choix représente le choix de la composante L. Deux méthodes s'offrent à lui : il peut d'abord choisir dans l'espace à 2 dimensions du carré un point de couleur (ce qui définit les composantes T et S de la couleur recherchée), puis choisir ensuite dans le rectangle le niveau de la composante L de la couleur choisie. L'opérateur peut aussi procéder en sens inverse : choisir d'abord une luminance L dans le rectangle de choix, puis choisir dans le carré de choix les composantes T et S. Dans les 2 méthodes, le carré présente des couleurs qui sont toujours les mêmes, quelle que soit la luminance L choisie dans le rectangle ; quant au rectangle, il est en réalité formé par la superposition de deux demi-rectangles de même hauteur : le rectangle supérieur contient les couleurs d'un certain segment OP_0 tandis que le rectangle inférieur contient celles du segment NP_0 ; le point P_0 dont il s'agit est le même pour ces deux demi-rectangles et il a des composantes S et L qui sont identiques et sont les mêmes que celles des couleurs présentes dans le carré de choix.
- Dans la première méthode, le point P_0 est choisi immédiatement en cliquant dans le carré de choix, et la couleur définitive désignée après le second choix (dans le rectangle) correspond à un point P qui a nécessairement les mêmes composantes S et L que le point P_0 , car le clic dans le carré (donc le choix de P_0) modifie immédiatement les couleurs qui apparaissent dans le rectangle, celles-ci ayant toutes les mêmes composantes S et L que le point P_0 .
- Dans la seconde méthode, le premier choix ne désigne par un point P_0 du *cône des couleurs* mais une certaine luminance L qui est celle d'un point R de couleur grise située sur l'axe NO ; quant aux couleurs qui apparaissent dans le carré après le premier clic (dans le rectangle), elles ont toutes la même luminance L que ce point R , et le choix représenté par le second clic (dans le carré de choix) fixe les composantes S et L qui associées à la composante L déjà choisie achèvent de définir la couleur recherchée.

- Dans les 2 méthodes, les couleurs représentées dans le carré de choix sont indépendantes des choix de l'utilisateur car elles correspondent toujours aux couleurs du cône d'axe GO et de sommet G , passant par (C), et le carré de choix n'est que la déformation de la surface de ce cône : le bord inférieur du carré correspond à une couleur unique qui est celle du point G (gris à 50 % de luminance), et le bord supérieur du carré correspond aux couleurs (de saturation $S = S_{max}$) du cercle situé à l'intersection de ce cône et du *cône des couleurs*.
- Dans le cas du second sélecteur de couleurs, les principes sont plus simples (mais l'utilisation n'est pas nécessairement plus pratique ...). Le rectangle de choix, dont les couleurs, invariables et saturées, sont celles du bord du cercle (C), est en général utilisé en premier et fixe la valeur de la teinte T. Ce premier clic modifie immédiatement les couleurs disponibles dans le carré de choix qui présente alors toutes les couleurs dont la teinte vaut T. Le choix d'un point dans ce carré fixe les valeurs des composantes S et L. Si l'opérateur n'est pas complètement satisfait de son choix, il peut l'affiner en cliquant à nouveau dans le rectangle, etc. Remarquons que les points du *cône de couleurs* qui correspondent aux couleurs disponibles dans le carré sont celles d'un certain triangle NOQ, Q étant un point situé sur le cercle (C). Le carré de choix est donc la déformation de ce triangle : le bord inférieur du carré correspond à une couleur unique qui est celle du point N (point noir), et le bord supérieur du carré correspond aux couleurs (de saturation $S = S_{max}$) du cercle (C).
- Habituellement, l'opérateur ne parvient pas du premier coup à choisir la *bonne* couleur et, quel que soit le sélecteur dont il dispose (Microsoft ou PhotoShop ou autre), il utilise donc en général successivement et en alternance, un clic dans le rectangle et un clic dans le carré. Quant aux valeurs numériques (TSL ou RVB), elles sont mises à jour comme il convient à l'occasion de chacun des clics de choix. L'opérateur peut aussi forcer telle ou telle valeur de ces composantes en les saisissant directement.

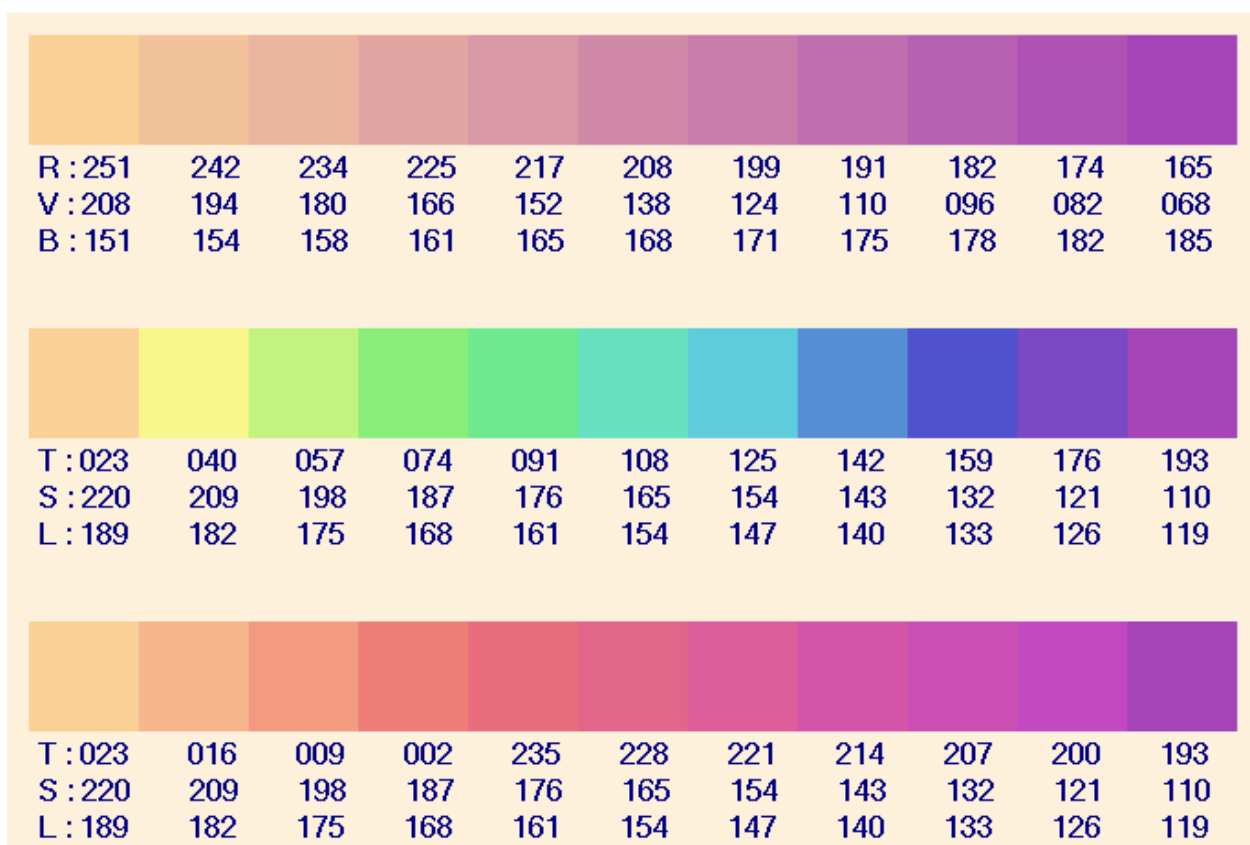
Dégradé de teintes

En s'inspirant des principes exposés ci-dessus, il est possible de définir différents dégradés de teintes. Donnons-en quelques exemples.

Voici d'abord une première série de 3 gammes de dégradés de 11 couleurs dont les 2 couleurs extrêmes (les couleurs n° 1 et 11) ont été choisies comme identiques.

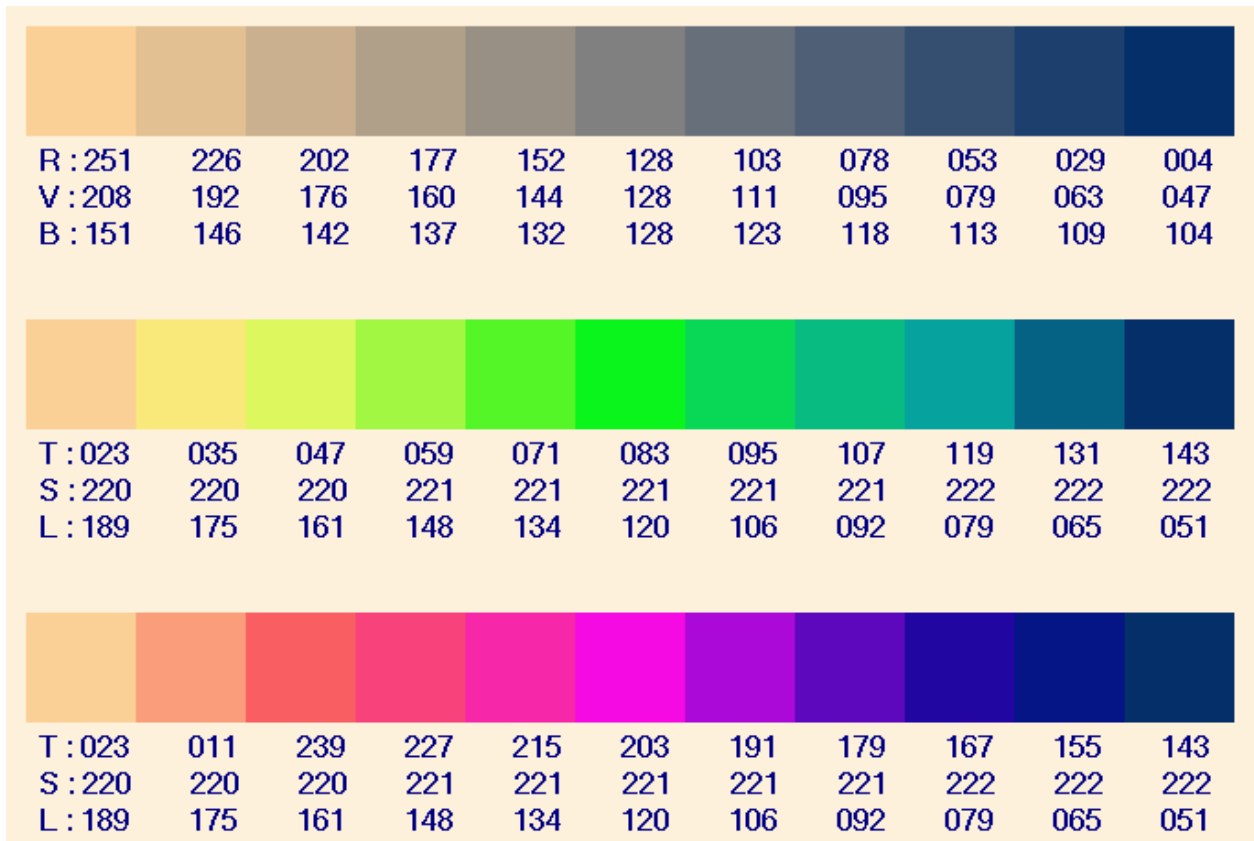
La couleur n° 1 correspond à la couleur choisie ci-dessus en exemple (rouge = 251, vert = 208, bleu = 151, teinte = 023, saturation = 220, luminance = 189), tandis que la couleur n° 11 est une couleur lilas sombre (rouge = 165, vert = 068, bleu = 185, teinte = 193, saturation = 110, luminance = 119).

- Le premier des 3 dégradés correspond à une progression linéaire des trois composantes rouge, vert et bleu (par exemple le vert décroît de 14 unités en passant d'une couleur à la suivante).
- Le second dégradé correspond à une progression linéaire des trois composantes teinte, saturation et luminance (par exemple la teinte croît de 17 unités en passant d'une couleur à la suivante).
- Enfin, le troisième dégradé correspond aussi à une progression linéaire des trois composantes teinte, saturation et luminance mais la teinte étant par nature une grandeur définie modulo 240, sa progression décroît linéairement de gauche à droite de 7 unités à chaque changement de couleur, mais au moment du passage à des valeurs négatives, il est ajouté 240 au résultat pour éviter la valeur négative -005 (qui devient donc 235) !



Ces 3 dégradés présentent des qualités et des défauts différents : le premier est celui dont l'aspect est le plus « régulier », mais les couleurs centrales sont un peu trop désaturées ; le second est celui qui donne les couleurs les plus vives mais il paraît assez peu « régulier », les couleurs n° 4, 5 et 6 étant trop voisines, de même que les couleurs 9 et 10 ; enfin le troisième présente un aspect intermédiaire : moins régulier mais plus vif que le premier, il est moins irrégulier que le second mais les teintes successives n° 5 à 10 ne sont pas assez distinctes.

Voici maintenant une seconde série de 3 gammes de dégradés de 11 couleurs aux 2 couleurs extrêmes identiques. Cette fois, la couleur n° 1 est la même que précédemment (rouge = 251, vert = 208, bleu = 151), mais la dernière a été choisie comme la couleur complémentaire exacte de celle-ci (rouge = 255 - 251, vert = 255 - 208, bleu = 255 - 151).



Les qualités et défauts signalés ci-dessus se retrouvent sur cette nouvelle gamme de dégradés, mais les défauts se trouvent amplifiés : la couleur n° 6 du premier dégradé est complètement désaturée (c'est un gris parfait de luminance 128), les teintes n° 5 et 6 du second sont très proches, les teintes n° 3 à 5 du troisième sont très proches et l'ensemble manque nettement de régularité.

On voit donc que, si l'on veut obtenir des gammes de dégradés à la fois régulières, peu ambiguës et fortement saturées, il sera généralement nécessaire de procéder par ajustements successifs, par exemple en panachant les méthodes utilisées ci-dessus.

Formules de changement de système de codage

RVB vers TSL

Voici maintenant les formules mathématiques qui régissent le passage des coordonnées R, G, B d'une couleur donnée aux coordonnées T, S, L de la même couleur, et réciproquement (les formules ci-dessous correspondent notamment sous Windows au codage utilisé dans l'utilitaire Paint livré avec les divers systèmes d'exploitation de Microsoft tournant avec des processeurs en 32 bits) :

Passage de R, V, B à T, S, L

- Calcul de la teinte T :

On doit d'abord calculer les 3 grandeurs préalables suivantes :

$$M = \max(R, V, B) \text{ (la plus grande des 3 composantes)}$$

$$m = \min(R, V, B) \text{ (la plus petite des 3 composantes)}$$

$$\Delta = M - m \text{ (l'amplitude, c'est-à-dire la différence entre } M \text{ et } m \text{)}$$

puis la valeur de T_0 qui se calcule ainsi :

- si R est supérieur à V et à B : $T_0 = (V - B)/\Delta + 0$
- si V est supérieur à B et à R : $T_0 = (B - R)/\Delta + 2$
- si B est supérieur à R et à V : $T_0 = (R - V)/\Delta + 4$

et enfin, en déduire la teinte T :

$$T = \frac{1}{6} (T_0 * T_{max}) \text{ modulo } T_{max}, \text{ avec } T_{max} = 240$$

- Calcul de la saturation S :

$$\text{- si } M + m < V_{max} : S = \frac{\Delta}{M + m} S_{max}$$

$$\text{- si } M + m \geq V_{max} : S = \frac{\Delta}{2 V_{max} - (M + m)} S_{max},$$

avec $V_{max} = 255$ et $S_{max} = 240$.

- Calcul de la luminance L :

$$L = \frac{M + m}{2 V_{max}} L_{max}, \text{ avec } L_{max} = 240.$$

Remarques : les formules donnant la valeur de la teinte T sont universelles ; en revanche, il existe de nombreuses variantes pour le calcul de la saturation S et de la L mais elles respectent toutes les règles suivantes :

- si R, V et B sont tous égaux, alors T est égal à 160 (dans Paint) et S est égal à 0 (teinte grise complètement désaturée), et réciproquement ;
- si R, V et B sont tous égaux à 0, alors L est égal à 0 (noir absolu), et réciproquement ;
- si R, V et B sont tous égaux à V_{max} , L est égal à L_{max} (blanc parfait), mais la réciproque n'est pas toujours vraie.

Voici donc quelques **variantes** :

- **pour la luminance** :

$$L = \frac{L_{max}}{V_{max}} M$$

$$L = \frac{L_{max}}{V_{max}} (0.239 * R + 0.686 * G + 0.075 * B)$$

$$L = \frac{L_{max}}{V_{max}} (0.300 * R + 0.590 * G + 0.110 * B)$$

- **pour la saturation** :

$$S = S_{max} \frac{\Delta}{M}.$$

Note : dans PhotoShop, les formules utilisées sont, pour la luminance, la première des 3 variantes citées ci-dessus, et, pour la saturation, la variante unique écrite ci-dessus.

TSL vers RVB

Passage de T, S, L à R, V, B

- Calculs préalables : on doit commencer, comme dans le cas précédent, par calculer les 4 grandeurs intermédiaires T_0, M, m et Δ :

Sachant que l'on a : $V_{max} = 255$ et $T_{max} = L_{max} = S_{max} = 240$, on calcule d'abord :

$$T_0 = \frac{6 T}{T_{max}}, \text{ puis}$$

$$\text{- si } L < L_{max}/2, \text{ on calcule : } M = V_{max} \frac{L}{L_{max}} \left(1 + \frac{S}{S_{max}}\right)$$

et

$$m = V_{max} \frac{L}{L_{max}} \left(1 - \frac{S}{S_{max}}\right)$$

- si $L \geq L_{max}/2$, on calcule : $M = V_{max} \left[\frac{L}{L_{max}} \left(1 - \frac{S}{S_{max}} \right) + \frac{S}{S_{max}} \right]$
 et
 $m = V_{max} \left[\frac{L}{L_{max}} \left(1 + \frac{S}{S_{max}} \right) - \frac{S}{S_{max}} \right]$,

d'où l'on peut tirer : $\Delta = M - m$.

Il devient dès lors possible de déterminer les valeurs des 3 composantes R , V , B

Note : Dans le cas du logiciel Paint Shop Pro, T_0 se calcule comme ci-dessus. Pour les 3 autres grandeurs intermédiaires, on doit utiliser successivement les trois formules suivantes qui permettent de calculer M , puis Δ et enfin m :

$$M = V_{max} \left[\frac{L}{L_{max}} \right] \text{ puis } \Delta = M \left[\frac{S}{S_{max}} \right] \text{ et enfin } m = M - \Delta$$

- Calcul des trois composantes R , V , B :

Les formules à utiliser dépendent de la valeur de la partie entière de T_0 :

Soit : $i = E(T_0)$

- si $i = 0$: $B = m$
 $V = m + T_0 \Delta$
 $R = M$

- si $i = 1$: $B = m$
 $R = m + (2 - T_0) \Delta$
 $V = M$

- si $i = 2$: $R = m$
 $B = m + (T_0 - 2) \Delta$
 $V = M$

- si $i = 3$: $R = m$
 $V = m + (4 - T_0) \Delta$
 $B = M$

- si $i = 4$: $V = m$
 $R = m + (T_0 - 4) \Delta$
 $B = M$

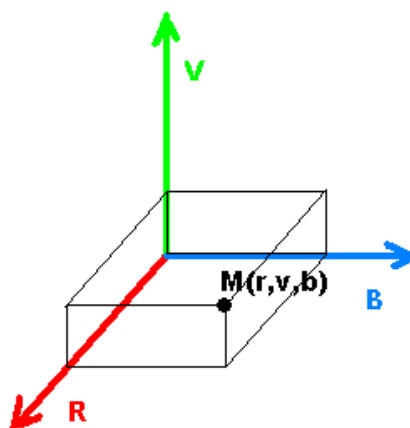
- si $i = 5$: $V = m$
 $B = m + (6 - T_0) \Delta$
 $R = M$

Généralisation du modèle de couleur

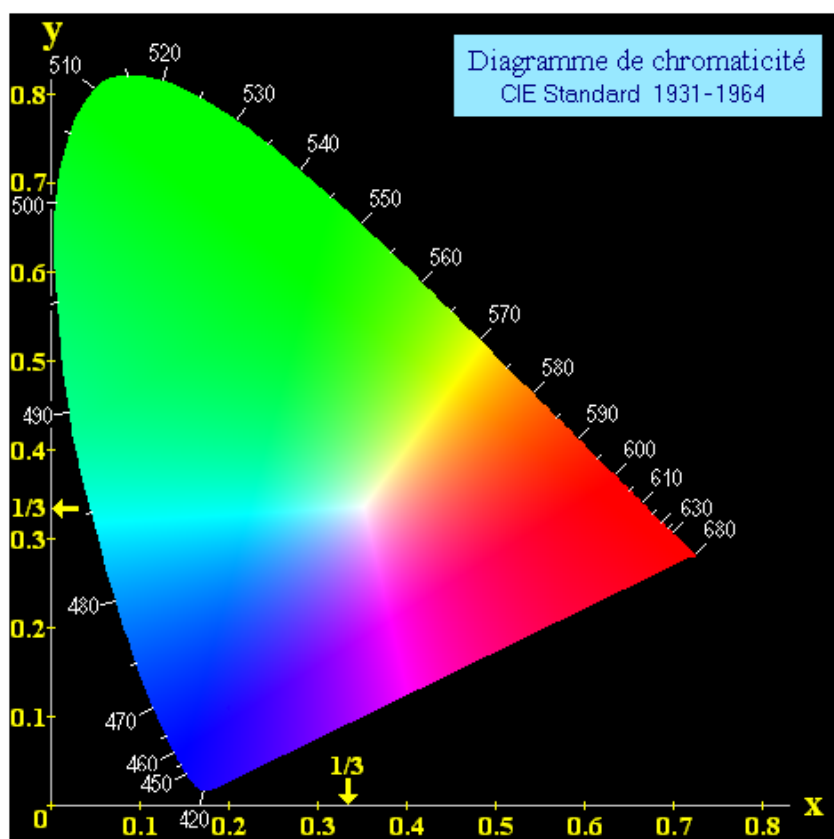
Considérons l'espace à 3 dimensions basé sur les composantes

- Rouge,
- Vert, et
- Bleu

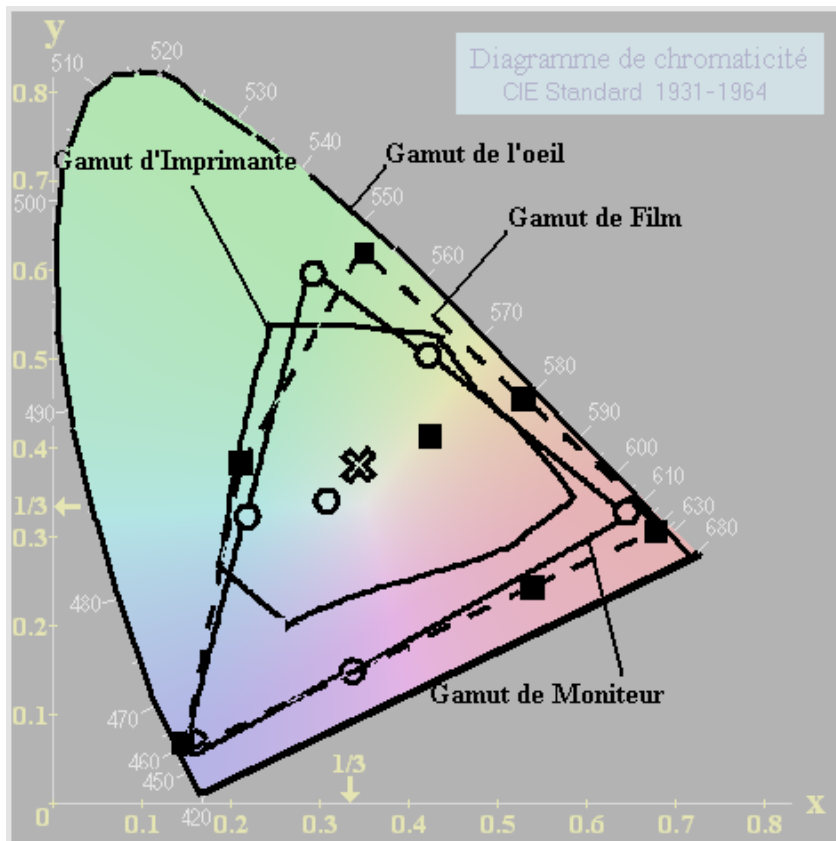
où tous les points ont une position définie dans un repère orthonormé par les 3 coordonnées (r,v,b).



L'œil humain n'est capable de percevoir qu'une partie de cet espace. L'ensemble des points de cet espace qui correspond aux possibilités visuelles de l'œil humain forme un volume convexe ou patatoïde, qui, projeté sur un plan convenablement orienté, donne une figure en forme de fer à cheval comme le montre l'image suivante, où les nombres à 3 chiffres, placés près du bord du fer à cheval, représentent les valeurs en nanomètres des longueurs d'onde associées aux couleurs monochromatiques saturées de l'arc-en-ciel :



Ce patatoïde correspond à ce que l'on appelle le gamut de l'œil humain. La figure ci-dessous, qui représente pour chaque type de support d'image (film, moniteur d'ordinateur, imprimante) un gamut distinct (et toujours plus réduit que celui de l'œil humain) illustre bien la pauvreté de certains supports en termes de rendu réel des couleurs.

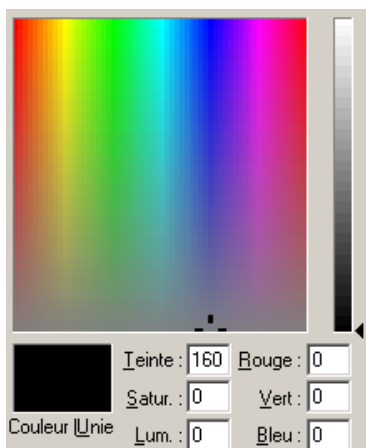


Maintenant, imaginons le cube de unitaire où $M=(255,255,255)$. Suivant ce cube, M sera la couleur blanche tandis que $(0,0,0)$ est noir. Si on se positionne maintenant suivant cet axe $[OM]$ depuis le point M , on observe

- un disque autour de ce point dont la couleur passe du Rouge, puis au Vert au Bleu puis revient au Rouge (si on fait un tour complet) avec tous les intermédiaires.
- suivant les rayons de ce disque, la couleur est pastel vers le centre et saturée vers le bord.
- suivant l'axe $[OM]$, on a une couleur qui est claire en allant vers M et sombre à l'allant vers O .

On a retrouvé ainsi le repère TSL ou HSL.

Une façon de représenter ce repère est de projeter le rayon de saturation et l'angle de teinte sur un cylindre qui est déplié suivant sa hauteur, on obtient le schéma suivant :



La transformation RVB/TSL est une simple transformation de repère qui peut être réalisée par une série de rotations et de translations combinées avec un passage d'un système de coordonnées cartésiennes à un système de Coordonnées polaires ; Les formules données au chapitre précédent correspondent à une transformation de ce type (parmi d'autres possibles).

On constate à partir de maintenant que la représentation dite en « RVB » ne peut proposer qu'un gamut réduit des possibilités :

- soit le cube propose des combinaisons inaffichables ou invisibles,
- soit le cube est inscrit dans le patateïde.

Le choix pratiqué a été le second. On a donc cherché d'autres formes de repères qui permettent d'agrandir le gamut.

Par exemple, si on utilise pour r, v et b des réels au lieu d'entiers, on peut obtenir un dégradé de teintes parfait car continu^[1].

Notons ainsi qu'en augmentant le nombre de bits par pixels, prenons par exemple 96 bits par pixels, chacune des composantes R,V,B est codée sur 32 bits : on obtient $2^{32} = 4\,294\,967\,296$ valeurs pour chaque composante (au lieu des 256 valeurs du codage 8 bits), avec un total de $2^{96} = 79 \times 10^{27}$ couleurs (au lieu des 16,7 millions du codage 8 bits/composante) : on peut alors faire "entrer la patateïde dans le cube", donc toutes les couleurs visibles peuvent être codées (et même aussi d'autres couleurs, telles que les ultraviolets ou les infrarouges, invisibles par l'œil mais visibles par des appareils spéciaux) : cf Note ^[2].

Toutefois, même si les ordinateurs performants du XXI^e siècle peuvent sans problème gérer ces très grands nombres de couleurs, les dispositifs techniques (écrans, imprimantes, vidéoprojecteurs, hologrammes, etc) ne peuvent pas actuellement (en 2012) les restituer fidèlement.

Notons aussi que le recours à une échelle logarithmique plutôt que linéaire permet d'obtenir plus de dynamique sur une zone choisie.

Par ces artifices, il devient dès lors possible de disposer de plus d'informations que le support (voire l'œil) ne peut en afficher.

Cette catégorie de procédés appartient aux images dites HDR.

Notes et références

[1] Il est bien entendu impossible de représenter pratiquement davantage qu'un nombre *fini* de réels, car chacun ne peut être défini que par un algorithme (par exemple e, pi, racine de 2...), mais on peut à défaut utiliser des *rationnels* également nommés nombres fractionnaires, et dont on peut choisir le dénominateur aussi grand qu'on le désirera

[2] Le système RVB ne peut fournir toutes les couleurs visibles que si R prend des valeurs négatives et positives; toutefois si on applique la transformation (d'abord avec des nombres réels) :

$$216 = 6 \times 6 \times 6 = 6^3$$

UNIQ-math-1-d6937a40dabe54f5-QINU

$$L = \frac{L_{max}}{2}$$

alors toutes les couleurs visibles (ainsi que les UV et IR) sont codées en respectant $0 < X < X_{max}$, et $0 < Y < Y_{max}$, et $0 < Z < Z_{max}$; de plus cette transformation est conforme à la norme CIE_XYZ de la Commission Internationale de l'Éclairage.

Afin de coder par des nombres entiers positifs chaque composante (Xn,Yn,Zn) sur (n) bits (donc chaque pixel sur (N = 3*n) bits), on effectue la discrétisation suivante où round(t) signifie : nombre entier le plus proche de (t) :

(C)

L_{max}

S_{max}

$N = 3*n$: donc (C) couleurs (dont la totalité des couleurs visibles) sont ainsi codées avec des nombres entiers positifs.

Exemple 1 : n = 21 bits donne N = 63 bits (pour obtenir la rapidité de calcul avec les ordinateurs 64 bits), d'où $2^{63} = 9\,223\,372\,036\,854\,775\,808$ couleurs avec $2^{21} = 2\,097\,152$ valeurs entières positives pour chacune des 3 composantes.

Exemple 2 : n = 32 bits donne N = 96 bits, d'où $2^{96} = 79 \times 10^{27}$ couleurs avec $2^{32} = 4\,294\,967\,296$ valeurs entières positives pour chacune des 3 composantes.

Sources et contributeurs de l'article

Codage informatique des couleurs *Source:* <http://fr.wikipedia.org/w/index.php?oldid=96264071> *Contributeurs:* (:Julien:), Akeron, Akiry, Anne Bauval, Antabuse, BlaX, Blue Prawn, Briling, Cdang, Charles Dyon, Christophe.moustier, Creasy, Crob, Crouchineki, Céréales Killer, Dadr, DarkBaboon, David Berardan, Dhatier, Epommate, Fighto, Flo, Freddo, Freewol, Herr Satz, Jaymz Height-Field, Jef-Infojef, Le sotré, Leag, Ltrlg, Maloq, Melkor73, MetalGearLiquid, Michka B, Neseb, Nicolas Ray, Nono64, Orthomaniaque, Papy77, Pautard, Phe, Piku, Pmx, PoM, Quasar, Ripounet, Romainhk, Romanc19s, Ropa, Sbgodin, Ssire, Stanlekub, Stéphane33, Sylvainm86, TP, 64 modifications anonymes

Source des images, licences et contributeurs

Image:Carnation.png *Source:* <http://fr.wikipedia.org/w/index.php?title=Fichier:Carnation.png> *Licence:* Public Domain *Contributeurs:* Papy77

Image:ColorsCone.png *Source:* <http://fr.wikipedia.org/w/index.php?title=Fichier:ColorsCone.png> *Licence:* Public Domain *Contributeurs:* Author:Papy77Papy77

Image:Trois_dégradés1.png *Source:* http://fr.wikipedia.org/w/index.php?title=Fichier:Trois_dégradés1.png *Licence:* Public Domain *Contributeurs:* Papy77

Image:Trois_dégradés2.png *Source:* http://fr.wikipedia.org/w/index.php?title=Fichier:Trois_dégradés2.png *Licence:* Public Domain *Contributeurs:* Papy77

Image:Codage RVB.PNG *Source:* http://fr.wikipedia.org/w/index.php?title=Fichier:Codage_RVB.PNG *Licence:* GNU Free Documentation License *Contributeurs:* Original uploader was

Image:Image-Gamut_couleurs.png *Source:* http://fr.wikipedia.org/w/index.php?title=Fichier:Image-Gamut_couleurs.png *Licence:* Public Domain *Contributeurs:* Papy77

Image:Gamuts - Fer à Cheval.PNG *Source:* http://fr.wikipedia.org/w/index.php?title=Fichier:Gamuts_-_Fer_à_Cheval.PNG *Licence:* GNU Free Documentation License *Contributeurs:* Original uploader was

Image:Codage_HSL.png *Source:* http://fr.wikipedia.org/w/index.php?title=Fichier:Codage_HSL.png *Licence:* Public Domain *Contributeurs:* Papy77

Licence

Creative Commons Attribution-Share Alike 3.0
[//creativecommons.org/licenses/by-sa/3.0/](http://creativecommons.org/licenses/by-sa/3.0/)