

# La phylogénie des images dans les réseaux sociaux

---



## Étude bibliographique

Master *Sciences et Technologies*,  
Mention *Informatique*,  
Parcours IMAGINA

**Auteur**

Noé Le Philippe

**Superviseurs**

William Puech  
Christophe Fiorio

**Lieu de stage**

Équipe ICAR - LIRMM UM5506 - CNRS, Université de Montpellier

## Résumé

Ce stage de master.

---

## Abstract

This master thesis.

---

# Table des matières

<b>Table des matières</b>	<b>iii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Near-duplicate images (NDI) . . . . .	2
1.2 Arbre phylogénétique (Image Phylogeny Tree - IPT) . . . . .	2
1.3 Pourquoi se restreindre à la compression comme transformation ? . . . . .	4
<b>2 État de l'art</b>	<b>5</b>
2.1 Étude de l'arbre phylogénétique . . . . .	5
2.2 Analyse des recompression JPEG . . . . .	8
2.3 Distances entre distributions . . . . .	12
<b>3 Notre approche</b>	<b>13</b>
3.1 Principe . . . . .	13
3.2 La construction de l'arbre . . . . .	13
<b>4 Conclusion</b>	<b>16</b>
<b>Bibliographie</b>	<b>17</b>

## Introduction

La phylogénie, en sciences naturelles, est définie comme l'étude des relations de parenté entre êtres vivants. Et c'est exactement de cela qu'il s'agit dans le cas des images, l'étude des relations de parenté entre images. [17]

À l'ère du numérique et des réseaux sociaux, il n'a jamais été aussi simple de partager des idées et du contenu. À chaque partage cependant, l'information peut être modifiée plus ou moins fortement. Les images, puisque c'est là notre sujet d'étude, peuvent avoir subi un certain nombre de transformations et modifications avant d'être publiée sur les réseaux sociaux. C'est dans ce contexte que nous allons intervenir afin tenter de reconstituer la phylogénie des images. Il peut être difficile de différencier une image modifiée de l'originale, et de savoir laquelle est l'originale, alors que c'est crucial dans un monde où l'information peut être facilement falsifiée par tout un chacun. Les applications sont multiples et variées, et ne se cantonnent pas à la détection et la discrimination d'images altérées. Il est également possible de se servir de la phylogénie des images pour optimiser de l'espace de stockage en ne gardant que l'originale ou encore suivre la diffusion et l'évolution des idées sur les réseaux sociaux.

Dans ce chapitre nous commencerons par définir ce que sont les near-duplicates, puis nous détaillerons la notion d'arbre phylogénétique et enfin nous expliquerons pourquoi nous nous sommes restreint à la compression JPEG dans le cadre de cette étude. Nous ferons un état de l'art dans le chapitre 2 où présenterons les différentes approches pour calculer l'arbre de phylogénie, nous y traiterons également la détection des recompressions ou l'estimation du nombre de recompressions. Nous présenterons ensuite notre approche dans le chapitre 3, où un algorithme de reconstruction d'arbre sera détaillé, en plus du théorème qui sera la base de la suite du stage. Nous finirons par conclure chapitre 4.



FIGURE 1.1 – Exemple de quantifications sur l’histogramme d’une gaussienne avec des classes de largeur 1, issu de [8]

## 1.1 Near-duplicate images (NDI)

Nous travaillons sur un ensemble d’images, toutes similaires visuellement, et au milieu de cet ensemble d’images, nous devons décider quelle image est le parent de quelle autre, ou autrement dit, quelles images sont des **near-duplicates**. JOLY, BUISSON et FRÉLICOT [11] définissent la notion de near-duplicate comme suit :  $I_{n+1} = T(I_n)$ ,  $T \in \mathcal{T}$  où  $I_n$  est l’image parent,  $I_{n+1}$  est l’image à la génération  $n + 1$ , l’image enfant et  $\mathcal{T}$  est un ensemble de transformations autorisées,  $I_{n+1}$  et  $I_n$  sont alors des NDI. Dans le cas général,  $\mathcal{T} = \{\text{resampling, cropping, affine warping, color changing, lossy compression}\}$ , la figure 1.1 montre un exemple de near-duplicates. Le *resampling*, ou rééchantillonnage en français revient à changer le nombre de pixels de l’image, le *crop* est illustré figure 1.1b, *affine warping* englobe tout ce qui peut être translation ou rotation, *color changing* concerne tout ce qui va changer la couleur, comme le changement de contraste par exemple, ou le passage au noir et blanc (figure 1.1a), et enfin *lossy compression* est la compression avec perte, une dégradation de l’image pour réduire son poids. Notons l’utilisation du terme *transformations autorisées*. Ce terme est double, d’une part, il place une limite arbitraire dans la force de la transformation, par exemple une image croquée à plus de 10% pourra ne pas être considérée comme un near-duplicate, et d’autre part, il permet de restreindre l’espace des transformations possibles. Ces transformations peuvent évidemment se composer, et une image enfant peut être le résultat de plusieurs transformations.

**Note.** Nous utiliserons les notions relation parent-enfant et relation de parenté de manière interchangeable dans le reste de ce rapport, mais c’est bien d’une relation parent-enfant qu’il s’agit.

## 1.2 Arbre phylogénétique (Image Phylogeny Tree - IPT)

L’arbre phylogénétique est l’arbre représentant les relations de parenté entre les différentes images. Il est extrait d’un ensemble de NDI, et constitue l’objectif final de l’application. La figure 1.2 illustre la construction de l’IPT à partir d’un ensemble de NDI. Le passage d’une génération à l’autre, autrement dit d’un noeud à son fils, se fait à travers la transformation  $I_{n+1} = T(I_n)$ , ainsi, une image et son parent sont des NDI, alors qu’une image  $I_{n+1,i}$  et sa soeur  $I_{n+1,j}$ ,  $i \neq j$  ne le sont pas (figure 1.3).

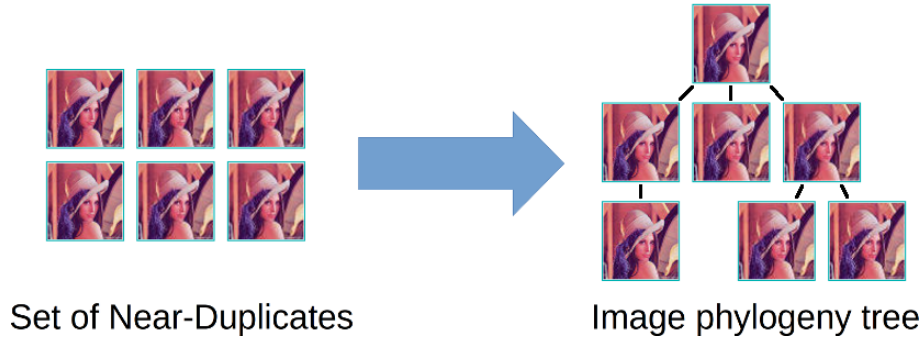


FIGURE 1.2 – Passage d'un ensemble de NDI à un arbre phylogénétique

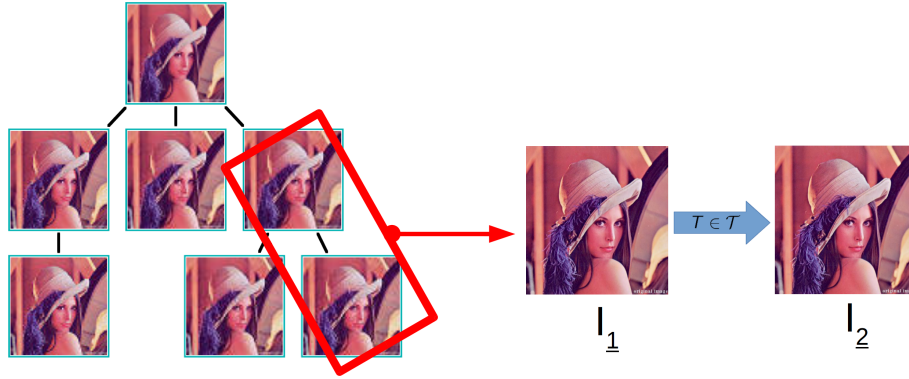


FIGURE 1.3 – Passage d'une image parent à l'image enfant

La reconstruction de l'arbre se concentre autour de deux problèmes principaux. Le premier est l'identification de la racine, et le second est l'estimation du reste de l'arbre, et en particulier positionner précisément chacune des images dans leur génération respective. Il est en effet critique d'identifier correctement la racine. Prenons par exemple un des cas d'utilisation de l'IPT, la détection d'altération d'images. L'idée est que pour un ensemble d'images, plus la génération est proche de la racine, c'est à dire plus  $n$  est petit, moins l'image a subi de transformations, et donc moins elle est altérée. Avec comme cas particulier  $n = 0$ , l'image est alors la génération 0 : la racine. Notons que si la racine est mal identifiée, une image pourra être à tort marquée comme altérée. on déduira, à tort, qu'une image n'a pas été altérée. Il n'est pas toujours garanti que la totalité des images de l'arbre original soit présente, l'identification précise de la génération peut alors se révéler complexe. Certaines transformations peuvent être mineures, et difficile à détecter, dans ce cas deux images de deux génération proches  $n + i$  et  $n + j$ , peuvent être identifiées à tort comme appartenant à la même génération. En conclusion, il est clair que l'arbre ne sera qu'une estimation, et sauf dans de cas idéaux, ne sera pas l'arbre original.

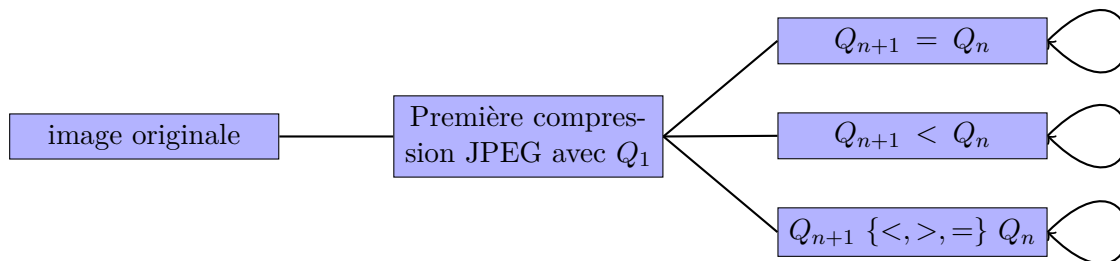


FIGURE 1.4 – Exemple des différents scénarios de recompression

### 1.3 Pourquoi se restreindre à la compression comme transformation ?

Dans le cadre du stage, l'espace des transformations sera réduit à  $\mathcal{T} = \{\text{lossycompression}\}$  avec JPEG comme compression. L'étude et la détection des recompressions est en effet à sujet suffisamment vaste et complexe pour que cela soit la seule transformation étudiée dans le cadre de ce stage. C'est de plus un sujet qui n'a à notre connaissance pas été traité dans le cadre de la phylogénie des images, et qui est largement étudié dans le domaine du forensic. L'étude et la détection des recompressions est en effet un sujet vaste, la figure 1.4 détaille l'étendue du problème. La compression JPEG sera détaillée chapitre 2 section 2.2, aussi la seule caractéristique dont nous parlerons maintenant est le facteur de compression  $Q$ .  $Q \in [1..100]$  et plus  $Q$  est grand, plus l'image sera de bonne qualité, mais moins elle sera compressée. Nous sommes donc en présence de trois cas pour les recompressions successives : le premier est le cas où tous les  $Q$  successifs sont égaux, lors du second cas, l'image est dégradée à chaque recompression, et dans le troisième cas, nous n'avons aucune idée sur la manière dont l'image a été compressée par rapport à son parent, elle peut avoir été recompressée avec la même qualité, ou une meilleure qualité, ou encore une moins bonne qualité.

## État de l'art

### 2.1 Étude de l'arbre phylogénétique

#### La Visual Migration Map (VMM)

C'est à notre connaissance, le premier article concernant vraiment notre sujet. Kennedy et al. [13] proposent une approche permettant d'automatiquement détecter la manière dont une image a été éditée ou manipulée, et d'en extraire des relations parent-enfant entre les images. Il vont construire à partir de l'estimation de ces transformations une Visual Migration Map (VMM) (voir Fig. 2.1) qui est en fait notre arbre de phylogénie.

Ils partent du principe que les transformations sont directionnelles, c'est à dire qu'il n'est possible d'aller que d'une image moins transformée vers une image plus transformée. Ainsi, ils vont tenter d'estimer la direction de chaque transformation entre deux images  $I_1$  et  $I_2$  (sachant que  $\mathcal{T} = \{\text{scaling, cropping, grayscale, overlay, insertion}\}$ ). Trois scénarios sont alors possibles : si toutes les transformations sont dans le même sens, l'image fille est alors celle vers qui pointent les transformations, si les transformations sont dans des sens contraires, les images sont sûrement des soeurs, elles n'ont en tous cas pas de relation parent-enfant, et enfin si aucune transformation n'a été détectée, c'est que soit les images sont identiques, soit qu'elles ne sont pas des near-duplicates. La figure 2.2 illustre ce principe.

Un graphe va ensuite être construit à partir des couples d'images pour lesquels une relation parent-enfant a été détectée. À noter qu'une relation parent-enfant ne veut pas forcément dire que c'est le parent direct mais plutôt un ancêtre. Ainsi, un noeud du graphe (une image) peut avoir plusieurs noeuds parents, pour finalement obtenir l'arbre désiré, seuls les chemins les plus longs sont conservés, comme on peut le voir figure 2.3.

#### Image Phylogeny Tree (IPT)

Tout comme l'approche présentée précédemment, DIAS, ROCHA et GOLDENSTEIN [6][7] proposent une approche basée sur le contenu de l'image. Cela consiste à mapper une image sur le domaine d'une autre, pour pouvoir les comparer, et ensuite estimer le coût de cette opération, cela fait comme hypothèse que si deux images sont dépendantes, alors il est possible d'obtenir l'une en appliquant une transformation à l'autre.



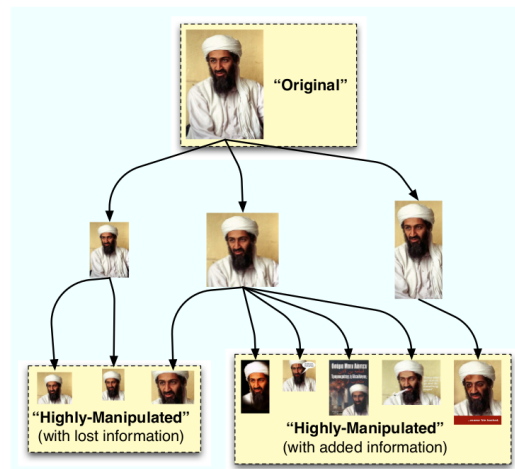


FIGURE 2.1 – Exemple de VMM, issu de [13]

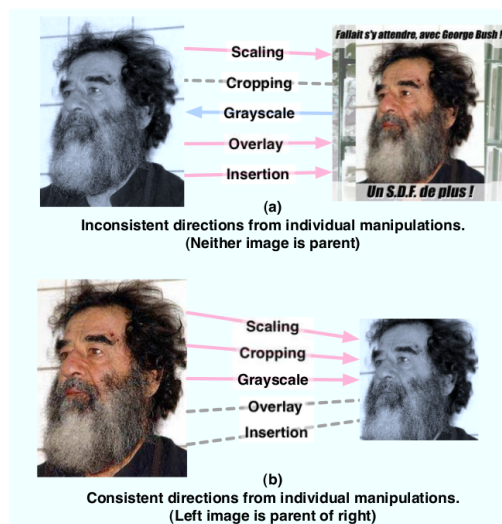


FIGURE 2.2 – Exemple de direction des transformations, issu de [13]

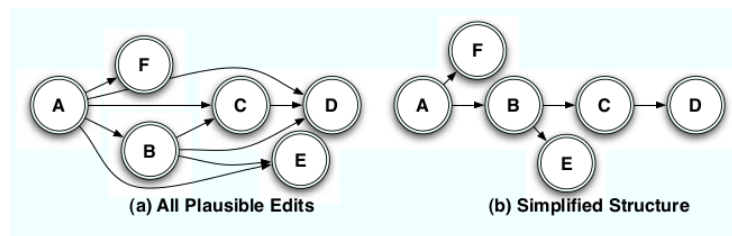


FIGURE 2.3 – Exemple de simplification de graphe, issu de [13]

Les images sont comparées à l'aide d'une *dissimilarity function*  $d(.,.)$  qui renvoie des petites valeurs lorsque les images sont proches (elles ont subi des transformations similaires). L'équation 2.1 détaille cette fonction.  $A$  et  $B$  sont les deux images qui vont être comparées, et  $T_{\vec{\beta}} \in \mathcal{T}$  est la transformation en train d'être estimée. La transformation la plus faible est gardée comme résultat, c'est la transformation la plus probable qu'a pu subir l'image. À noter que  $d(.,.)$  est asymétrique, c'est à dire que  $d(A, B) \neq d(B, A)$ , ce qui est parfaitement logique, en plus d'être nécessaire, sachant que les transformation sont directionnelles, comme expliqué précédemment.

$$d(A, B) = \min_{T_{\vec{\beta}}} \left| B - T_{\vec{\beta}}(A) \right|_{\text{comparison method}} \quad (2.1)$$

On voit donc bien que la problématique principale de cette méthode est de trouver une bonne méthode de comparaison. Les auteurs procèdent de la manière suivante :

- Trouver des points caractéristiques (SURF [1])
- Filtrer les points et estimer les paramètres de transformation affines tels que les translations, rotations, rééchantillonnages... avec RANSAC [9]
- Calculer la moyenne et la variance de chaque canal couleur de B pour normaliser les couleurs de A
- Compresser les résultats des deux étapes précédentes avec la table de quantification de B

La dissimilarité entre les deux images est enfin obtenue en utilisant la *minimum squared error* (MSE) entre les deux images dans le domaine spatial comme technique de comparaison pour Équation 2.1.

Ces quatre étapes ont servi à rendre les images comparables, en mappant l'une dans le domaine de l'autre, pour avoir des résultats pertinents.

$d(.,.)$  est donc appliquée à tous les couples d'images de l'ensemble, pour créer une *dissimilarity matrix*, une matrice de taille  $n \times n$  qui sera ensuite donnée comme entrée à leur algorithme de reconstruction d'arbre, Oriented Kruskal. Nous n'allons pas entrer dans les détails de cet algorithme, il est expliqué de manière exhaustive dans [7], et ayant une approche différente (Chap. 3), la reconstruction sera également différente.

En plus de proposer une approche pour reconstituer l'arbre de phylogénie, ils proposent une approche pour comparer deux arbres, et donc évaluer notre arbre reconstruit si on le compare avec la vérité terrain. Cela consiste en quatre métriques :

$$\textbf{Root} \quad R(IPT_1, IPT_2) = \begin{cases} 1 & \text{if } \text{Root}(IPT_1) = \text{Root}(IPT_2) \\ 0 & \text{Otherwise} \end{cases}$$

$$\textbf{Edges} \quad E(IPT_1, IPT_2) = \frac{|E_1 \cap E_2|}{n-1}$$

$$\textbf{Leaves} \quad L(IPT_1, IPT_2) = \frac{|L_1 \cap L_2|}{|L_1 \cup L_2|}$$

$$\textbf{Ancestry} \quad A(IPT_1, IPT_2) = \frac{|A_1 \cap A_2|}{|A_1 \cup A_2|}$$

**Root** est triviale, et renvoie si les racines sont identiques. **Edges** mesure le ratio de noeuds ayant le bon parent, **Leaves** est le ratio de feuilles correctes, et enfin **Ancestry** est le ratio d'ancêtres corrects jusqu'à la racine.

Ces métriques serviront à évaluer nos résultats dans la suite du stage.

## 2.2 Analyse des recompression JPEG

Notre but n'est pas vraiment de détecter si une image a été compressée plusieurs fois, nous sommes en effet quasi-certains que nos images auront été recompressées, puisque c'est précisément sur cela que nous travaillons, un ensemble de NDI, où  $\mathcal{T} = \{\text{lossy compression}\}$ . L'important est plutôt de savoir combien de fois, et à partir de qui l'image a été compressée, et donc pouvoir en déduire sa distance à la racine. La majorité des articles dans le domaine du forensic ne se concentrent que sur une image, pour en extraire le maximum d'informations possible. Ce n'est pas exactement notre cas, puisque les informations pertinentes pour nous ne sont pas dans l'image directement, mais plutôt dans ses relations avec les autres. Il nous a cependant paru important de traiter l'aspect forensic du problème, et se renseigner sur les différentes techniques, qui bien que créées pour un autre cas d'utilisation, peuvent, sinon s'adapter, au moins nous donner des pistes.

### Qu'est ce que JPEG ?

Une rapide introduction sur ce qu'est le JPEG et son fonctionnement s'impose, nous ne parlerons cependant que du mode de compression avec perte, en laissant de côté son mode de compression sans perte, peu intéressant en plus de n'être virtuellement jamais utilisé. Pour de plus amples détails, le lecteur se dirigera vers [18].

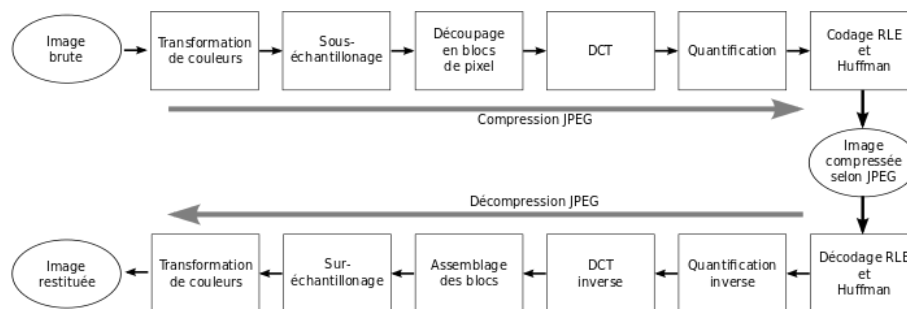


FIGURE 2.4 – Étapes de compression et décompression, issu de [12]

Fig. 2.4 liste toutes les étapes permettant de passer d'une image non compressée à une image compressée pour finir par la décompresser, mais les étapes sont sommairement :

- L'image est convertie dans l'espace YUV
- Les canaux U et V sont sous-échantillonnés
- Chaque canal est découpé en blocs de  $8 \times 8$

- Une DCT est appliquée à chacun de ces blocs pour avoir une matrice de  $8 \times 8$  coefficients
- Chaque coefficient est quantifié selon la table de quantification (voir Fig. 2.5) correspondant au facteur de qualité  $Q \in \{1, 2, 3, \dots, 100\}$  et arrondi
- Le tout est ensuite compressé à l'aide d'un codage entropique

C'est surtout l'étape de quantification qui va réduire la quantité d'information, et permettre de réduire la taille du fichier. Cette quantification, si elle est trop agressive va faire apparaître des artefacts de bloc, des blocs  $8 \times 8$  visibles (voir Fig. 2.6). C'est ce qui caractérise le JPEG, et qui permet de détecter un certain nombre de choses, comme l'altération d'image [3], les doubles compressions [2] ou encore dans notre cas les relations de parenté.

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

FIGURE 2.5 – Exemple de table de quantification, issu de [12]

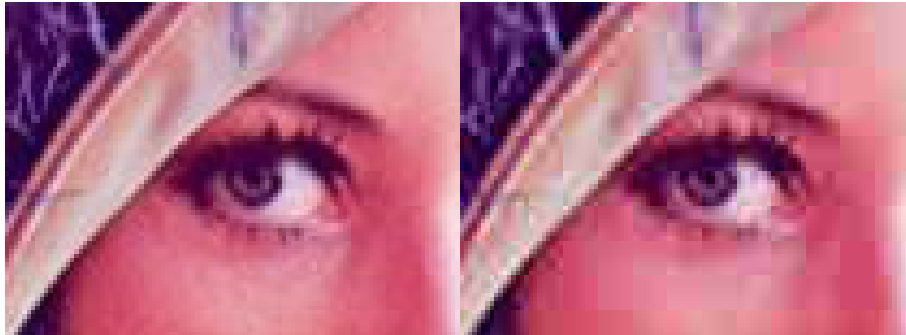


FIGURE 2.6 – Artefacts de blocs, à gauche avec  $Q=90$ , à droite avec  $Q=20$

### Estimation de la matrice de compression primaire dans les images JPEG doublement compressées

LUKÁŠ et FRIDRICH [15] proposent dans leur approche non seulement de détecter si une image est doublement compressée, mais également de d'estimer les paramètres de la première compression. Pour bien comprendre la suite de leurs travaux, il convient d'expliquer plus avant l'étape de quantification de JPEG.

La quantification se fait de la manière suivante :

$$F^*(u, v) = \left\lfloor \frac{F(u, v) + \left\lfloor \frac{Q(u, v)}{2} \right\rfloor}{Q(u, v)} \right\rfloor \quad (2.2)$$

Et le calcul inverse :

$$F'(u, v) = F^*(u, v) * Q(u, v) \quad (2.3)$$

où  $F(u, v)$  est la valeur aux indices  $u$  et  $v$  dans la matrice  $8 \times 8$  de coefficients DCT et  $Q(u, v)$  est la valeur dans la table de quantification (Fig. 2.5) à ces mêmes indices.

On voit bien à partir des équations 2.2 et 2.3 que le résultat après décompression sera un multiple de  $Q(u, v)$  et que toutes les valeurs n'étant pas des multiples seront absentes.

La double compression est donc le fait de compresser deux fois une image, et donc de lui faire subir deux fois toutes les étapes de compression et décompression, avec tous les arrondis et troncage que cela peut comporter. On notera, pour la double compression,  $Q^1$  comme la matrice primaire, c'est à dire la table de quantification ayant servi à faire la première compression (inconnue donc), et  $Q^2$  comme la matrice secondaire, ou la table de quantification actuelle, disponible dans l'en-tête du fichier JPEG. Les auteurs limitent la double compressions aux cas où  $Q^1 \neq Q^2$ .

Lors de la première compression, les coefficients sont quantifiés avec  $Q^1$ , ce qui veut dire que les coefficients sont des multiples de  $Q^1$ . Cependant, lorsque l'image est repassée du domaine fréquentiel au domaine spatial grâce à la DCT inverse, un certain nombre d'arrondis et de troncages se produisent pour rester dans l'intervalle  $[0..255]$ . Lors de la seconde compression, les coefficients DCT sont calculés à partir des ces valeurs inexacts, et donc ne seront pas multiples de  $Q^1$ , mais se concentreront autour. Ces nouveaux coefficients DCT seront ensuite quantifiés pour former la nouvelle image, doublement compressée.

Les auteurs ayant laissé de côté le cas où  $Q^1 \neq Q^2$ , nous sommes en présence de deux cas :  $Q^1 > Q^2$  et  $Q^1 < Q^2$ . Chacun des cas à des artefacts distincts et reconnaissables. Fig. 2.7 montre un exemple de distributions ayant subi des quantifications, 2.7a n'a subi qu'une seule quantification, 2.7b correspond au cas où  $Q^1 > Q^2$  et 2.7c correspond au deuxième cas. Un autre cas est délaissé, c'est celui où  $Q^1$  est facteur  $Q^2$ , ce qui est assez logique, ils auront les mêmes multiples, et il sera donc impossible de détecter quoi que ce soit, du moins en utilisant la méthode proposée ici.

L'estimation de la première table de quantification se limite aux basses fréquences, les hautes fréquences étant plus fortement quantifiées (souvent jusqu'à 0), elles contiennent moins d'informations et donc rendent leur estimation difficile.

Leur technique consiste à se munir d'une série de candidats pour la matrice primaire, notés  $\{Q^{1,1}, \dots, Q^{1,n}\}$ . Ces tables de quantification viennent des tables utilisées dans les appareils photos, les téléphones portables ou les différentes implémentations de la compression JPEG. Parmi ces  $\{Q^{1,1}, \dots, Q^{1,n}\}$ , le meilleur candidat, et donc la table la plus probable sera la table de quantification pour laquelle la différence entre l'histogramme de l'image et l'histogramme de l'image compressée avec  $Q^{1,k}$  sera la plus faible.

## Détection des double compressions JPEG avec la même matrice de quantification

[10]

### Convergence des blocs lors de compressions successives

L'intérêt de la méthode proposée par CARNEIN, SCHÖTTLE et BÖHME [5] est l'estimation du nombre de compression JPEG qu'a pu subir l'image, une estimation qui va au-delà de deux ou

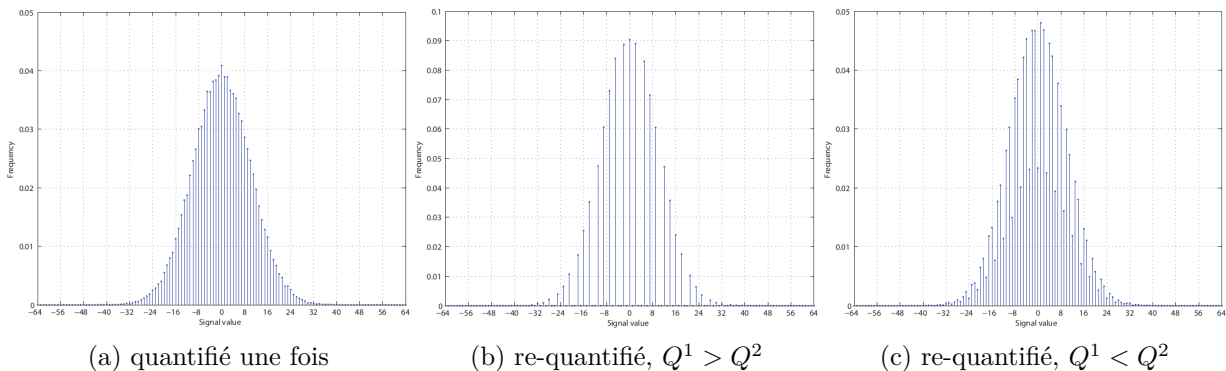


FIGURE 2.7 – Exemple de quantifications sur l’histogramme d’une gaussienne avec des classes de largeur 1, issu de [8]

trois compression successives [10][15], il est en effet possible d’aller jusqu’à plusieurs centaines de compressions. Cette approche se base sur la notion de blocs cycliques.

Lors de compressions successives, un phénomène appelé *convergence de bloc* peut être observé. Un bloc converge lorsque la valeur des pixels du bloc à la compression  $t$  est égale à la valeur des pixels à la compression  $t + 1$ , ce bloc est alors appelé stable [14]. Certains blocs cependant échappent à cet effet et exhibent un phénomène de cycle. C’est à dire que  $\text{valeur des pixels}(t) = \text{valeur des pixels}(t + n), n > 1$ .

Ces blocs sont nommés blocs compteurs. Ainsi, si un bloc de l’image a un cycle de longueur  $l = 9$ , on pourra savoir, modulo  $l$ , combien de compressions a subi l’image. Et avec plus de blocs, ayant chacun des  $l$  différents, il est possible d’aller bien plus loin dans le nombre de compressions. Prenons l’exemple de trois blocs pour lesquels  $l_1 = 2$ ,  $l_2 = 5$  et  $l_3 = 9$ . Le nombre maximum de compressions successives qu’il sera possible d’estimer est le *plus petit commun multiple* des ces longueurs, autrement dit,  $\text{ppcm}(2, 5, 9) = 90$ , soit 90 compressions successives, et pour l’exemple Fig. 2.8,  $\text{ppcm}(6, 7, 10, 12) = 420$ .

Les auteurs proposent deux approches pour les blocs cycliques. La première est de détecter les blocs cycliques à l’aide d’une recherche exhaustive sur l’ensemble des blocs et un certain nombre de compressions puis sélectionner les blocs intéressants. Cette méthode, bien qu’étant la plus simple, et ne modifiant pas l’image, peut cependant rencontrer des problèmes lorsqu’elle comporte un grand nombre de blocs plats, des blocs ayant tous les pixels de la même valeur, ces blocs ne peuvent en effet pas être des blocs cycliques, puisque tous les coefficients DCT sauf le premier sont nuls, et il peut alors être difficile de trouver des blocs ayant des cycles de longueur intéressante. La deuxième approche proposée est donc d’insérer des blocs artificiellement créés dans l’image (voir Fig. 2.8).

Cette méthode nécessite de prendre un certain nombre de précaution et requiert une protection (padding) autour des blocs insérés. L’algorithme de sur-échantillonnage est en effet parfois amené à utiliser les valeurs des pixels des blocs adjacents pour un meilleur rendu et provoquer un effet de débordement (spill over) des valeurs [4], ce qui modifie le bloc artificiellement créé (dans le vide) et donc modifie son cycle, rendant nulle son information.



FIGURE 2.8 – Exemple d'insertion de blocs, issu de [5]

Le principal inconvénient de cette méthode, si ce sont les blocs présents dans l'image qui sont utilisés, et que la technique d'insertion de blocs est laissée de côté, est qu'elle se restreint à des images ayant toutes le même facteur de qualité, et en particulier  $Q = 100$ , pour lequel les cycles sont les plus longs, et donc donnent le plus d'informations. C'est un cas spécifique pour notre cadre d'utilisation : les réseaux sociaux, où les facteurs de qualité de compression varient en fonction de l'application utilisée. Cette méthode est en effet plus orientée vers le tatouage et le traçage d'images.

## 2.3 Distances entre distributions

[16]

Même si notre approche, et donc pourquoi nous devons avoir la distance entre histogrammes ne sera expliqué qu'au chapitre suivant, nous aurons besoin de comparer des histogrammes, et c'est dans l'état de l'art que nous traiterons les différentes techniques.

Il convient également de définir la différence entre une distance et une divergence. Une distance, contrairement à une divergence, est symétrique, c'est à dire que  $distance(A, B) = distance(B, A)$ , et elle vérifie l'inégalité triangulaire.

Si dans notre cas la distance pourrait servir à estimer la proximité entre deux images, nous étudions les relations parent-enfant, relation qui n'est pas symétrique. L'utilisation d'une divergence est indispensable si nous voulons estimer la direction de cette relation.

## Notre approche

### 3.1 Principe

**Théorème.** *Pour tout couple d'images  $(A, B)$ , s'il n'est pas possible de prouver que  $A$  n'est pas le parent de  $B$ , alors il y a une relation parent-enfant entre  $A$  et  $B$ ,  $A \rightarrow B$ .*

*Démonstration.* Soit  $f(A, B)$  une fonction qui pour tout couple d'images  $(A, B)$  détecte à chaque fois qu'il est présent un marqueur prouvant qu'il n'y a pas de relation de parenté entre  $A$  et  $B$ . Si  $f(A, B)$  ne détecte rien alors c'est que  $A$  est le parent de  $B$ .  $\square$

### 3.2 La construction de l'arbre

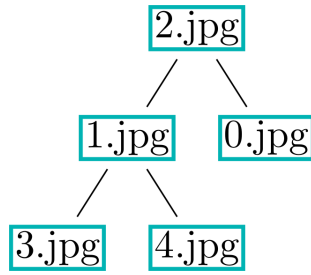
On va ainsi pour tout couple d'images de notre ensemble tenter de nier qu'il existe une relation de parenté, cela va permettre d'extraire une matrice binaire, de la taille de l'ensemble d'image, où une case à vrai indiquera une relation de parenté. À noter une fois de plus que ce n'est pas forcément un parent direct mais plutôt un ancêtre.

On peut noter plusieurs choses intéressantes de l'exemple Fig. 3.1. L'image  $I_2$  a toute sa colonne marquée à vrai, c'est à dire que c'est un parent commun à toutes les autres images, et sa ligne n'a que des 0, elle n'a donc aucun parent. On se sert de ce principe pour la reconstruction de l'arbre à partir de la matrice : une image qui n'a pas de parent est la racine. On peut également remarquer les colonnes où toutes les cases sont à 0, cela veut dire que ces images ne sont le parent de personne, et donc sont des feuilles.

#### L'algorithme de reconstruction

De ces quelques principes nous avons extrait un algorithme.





(a) Arbre de phylogénie

-	$I_0$	$I_1$	$I_2$	$I_3$	$I_4$
$I_0$	-	0	0	0	0
$I_1$	0	-	0	1	1
$I_2$	1	1	-	1	1
$I_3$	0	0	0	-	0
$I_4$	0	0	0	0	-

(b) Matrice de parenté

FIGURE 3.1 – Une arbre de phylogénie et sa matrice de parenté

**Data:**  $M$  a  $n \times n$  parentage matrix

**Result:** the root of the tree

```

1  $nextRoot \leftarrow$  row with min sum of elements;
2  $treeRoot \leftarrow nextRoot$ ;
3 forall the rows  $row$  of  $M$  do
4    $root \leftarrow nextRoot$ ;
5   mark  $root$  as done;
6   for  $i \leftarrow 0$  to  $n$  do
7      $row[i] \leftarrow 0$ ;
8     if sum of elements of  $row == 0$  then
9       | add  $i$  as child of  $root$ ;
10    end
11    if  $row$  has the smallest sum of elements and is not marked as done then
12      |  $nextRoot \leftarrow i$ ;
13    end
14  end
15 end
16 return  $treeRoot$ 

```

L'algorithme prend en entrée la matrice de parenté détaillée précédemment et retourne la racine de l'arbre. La philosophie de cet algorithme est qu'une image n'ayant aucun parent est la racine, et ce de manière récursive grâce à la structure d'arbre.

À chaque tour de boucle, une image est sélectionnée comme la racine (lignes 1 et 11) et nommée  $root$ , elle est retirée (ligne 7) des ancêtres des autres images si c'est un ancêtre. Si ces autres images n'ont plus d'ancêtre (ligne 8), c'est que  $root$  était le parent direct de l'image en train d'être traitée, cette image est donc ajoutée comme enfant de  $root$  (ligne 9). La ligne 5 permet de ne traiter qu'une fois chaque image comme racine potentielle.

Cet algorithme a une complexité de  $O(n^2)$ . Il y a deux boucles imbriquées, et si les sommes sont calculées une seule fois au début et mises à jour à chaque fois qu'un parent est enlevé, il n'y a pas de boucle supplémentaire augmentant la complexité.

Le but est donc de trouver la fonction permettant de détecter le marqueur prouvant qu'il n'y a pas de parenté.

Il nous a semblé important, même dans le cadre de l'étude bibliographique, d'élaborer cet algorithme. Il est en effet assez simple mais néanmoins indispensable à notre méthode, et aurait pu orienter différemment nos recherches et notre méthode s'il n'avait pas été imaginé. Le reste du stage sera consacré à élaborer la fonction énoncée précédemment, et c'est là que la majorité du travail se concentrera.

Cette approche a pour avantage de réduire l'évaluation d'un arbre de phylogénie à partir d'un ensemble d'images à l'évaluation binaire de parenté entre deux images.

---

## Conclusion

---

## Bibliographie

- [1] Herbert BAY et al. “Speeded-up robust features (SURF)”. In : *Computer vision and image understanding* 110.3 (2008), p. 346–359.
- [2] Tiziano BIANCHI et Alessandro PIVA. “Detection of nonaligned double JPEG compression based on integer periodicity maps”. In : *Information Forensics and Security, IEEE Transactions on* 7.2 (2012), p. 842–848.
- [3] Tiziano BIANCHI et Alessandro PIVA. “Image forgery localization via block-grained analysis of JPEG artifacts”. In : *Information Forensics and Security, IEEE Transactions on* 7.3 (2012), p. 1003–1017.
- [4] Matthias CARNEIN, Pascal SCHÖTTLE et Rainer BOHME. “Forensics of high-quality JPEG images with color subsampling”. In : *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*. IEEE. 2015, p. 1–6.
- [5] Matthias CARNEIN, Pascal SCHÖTTLE et Rainer BÖHME. “Telltale Watermarks for Counting JPEG Compressions”. In : *Proceedings of the Electronic Imaging 2016*. Publication status : Published. San Francisco, USA, 2016.
- [6] Zandoni DIAS, Anderson ROCHA et Siome GOLDENSTEIN. “First steps toward image phylogeny”. In : *Information Forensics and Security (WIFS), 2010 IEEE International Workshop on*. IEEE. 2010, p. 1–6.
- [7] Zandoni DIAS, Anderson ROCHA et Siome GOLDENSTEIN. “Image phylogeny by minimal spanning trees”. In : *Information Forensics and Security, IEEE Transactions on* 7.2 (2012), p. 774–788.
- [8] Xiaoying FENG et Gwenaél DOËRR. “JPEG recompression detection”. In : *IS&T/SPIE Electronic Imaging*. International Society for Optics et Photonics. 2010, 75410J–75410J.
- [9] Martin A. FISCHLER et Robert C. BOLLES. “Random Sample Consensus : A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography”. In : *Commun. ACM* 24.6 (juin 1981), p. 381–395. ISSN : 0001-0782. DOI : 10.1145/358669.358692. URL : <http://doi.acm.org/10.1145/358669.358692>.
- [10] Fangjun HUANG, Jiwu HUANG et Yun Qing SHI. “Detecting double JPEG compression with the same quantization matrix”. In : *Information Forensics and Security, IEEE Transactions on* 5.4 (2010), p. 848–856.

- [11] Alexis JOLY, Olivier BUISSON et Carl FRÉLICOT. “Content-based copy retrieval using distortion-based probabilistic similarity search”. In : *Multimedia, IEEE Transactions on* 9.2 (2007), p. 293–306.
- [12] *JPEG*. URL : <https://fr.wikipedia.org/wiki/JPEG>.
- [13] Lyndon KENNEDY et Shih-Fu CHANG. “Internet image archaeology : automatically tracing the manipulation history of photographs on the web”. In : *Proceedings of the 16th ACM international conference on Multimedia*. ACM. 2008, p. 349–358.
- [14] ShiYue LAI et Rainer BOHME. “Block convergence in repeated transform coding : JPEG-100 forensics, carbon dating, and tamper detection”. In : *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE. 2013, p. 3028–3032.
- [15] Jan LUKÁŠ et Jessica FRIDRICH. “Estimation of primary quantization matrix in double compressed JPEG images”. In : *Proc. Digital Forensic Research Workshop*. 2003, p. 5–8.
- [16] Marina A OIKAWA et al. “Distances in multimedia phylogeny”. In : *International Transactions in Operational Research* (2015).
- [17] *Phylogénie*. URL : <https://fr.wikipedia.org/wiki/Phylogen%C3%A8se>.
- [18] Gregory K WALLACE. “The JPEG still picture compression standard”. In : *Consumer Electronics, IEEE Transactions on* 38.1 (1992), p. xviii–xxxiv.