

Restauració d'imatges JPEG

Lluís Batlle

1 de desembre de 2008

Índex

1	Plantejament del problema	7
1.1	La codificació JPEG	7
1.2	Popularitat	8
1.3	Les pèrdues al JPEG	10
2	Qualitat de les imatges	17
3	Tècniques	21
3.1	Projecció sobre l'espai de quantització	21
3.2	Coeficients com a variables aleatòries	22
3.3	Segmentació i suavitzat	23
3.4	POCS - Projecció sobre conjunts convexos	24
3.5	Maximum A Posteriori	25
3.6	Altres mètodes de recuperació de plans	25
3.7	Recuperació del color basada en lluminositat	25
4	Programari	27
4.1	La cadena de descodificació	29
4.2	Desquantització i IDCT 2D de cada pla	31
4.3	Restaurador de pla	31
4.4	Escalador dels plans	32
4.5	Canvi de plans de color	34
5	Anàlisi dels mètodes implementats	37
5.1	Plans a molt poca qualitat	38
5.2	Efecte d'ones superior al de blocs	41
5.3	Dibuixos	41
6	Conclusions i treball futur	47
A	Transformada DCT	49

B Algorismes en detall	51
B.1 Manteniment de textures	51
B.2 Escalat dels plans de color ponderats amb la lluminositat . . .	52
Bibliografia	55

Introducció

La codificació d'imatges JPEG ofereix unes imatges bastant fidels amb molt pocs bits per píxel, reduint en bona mesura la mida dels fitxers d'imatge i els seus temps de transferència per xarxa. Per això ha estat molt popular en la publicació d'imatges per Internet, i ho continua sent en l'ambient domèstic degut a les càmeres digitals. Així doncs, creiem que qualsevol esforç a millorar la descodificació d'aquestes imatges té molt valor.

Ens hem decidit a analitzar i recopilar les nombroses tècniques de restauració de JPEG aparegudes a diferents publicacions científiques. Alhora, hem creat una aplicació informàtica lliure que implementa alguns dels algorismes més interessants de la nostra cerca, a més d'establir una base per al desenvolupament d'aquest tipus de tècniques.

Mitjançant aquest informe i el programa informàtic desenvolupat en paral·lel introduïm al lector al món de la pèrdua d'informació JPEG, donem a conèixer l'estat de l'art de la seva restauració, i proveïm als manipuladors d'imatges (de centres fotogràfics, d'impressió, etc.) amb les tècniques més prometedores d'aquest camp. Portem les idees de les publicacions científiques de laboratori als ordinadors personals de cases i centres de fotografia.

Al llarg del document trobarem varies imatges exemplificant soroll i restauracions, algunes d'elles difícils d'apreciar amb una impressora làser convencional. Per això us remetem a la versió electrònica d'aquest document per a poder apreciar les diferències entre imatges en detall.

Capítol 1

Plantejament del problema

1.1 La codificació JPEG

El 1986 es va crear un comitè per establir un estàndard de codificació d'imatges de to-continu (continuous tone), per exemple, fotografies. Aquest comitè tenia el nom de Joint Photographic Experts Group, i el 1992 la ISO i el CCITT van aprovar l'estàndard conegut arreu amb el nom de JPEG (ISO/IEC IS 10918-1 - ITU-T Recommendation T.81)¹. Van decidir tant la codificació de la imatge, efectivament amb les sigles del grup, i en paral·lel també es va decidir un format de fitxer que contindria la informació codificada (JFIF - JPEG File Interchange Format), ja que en informàtica sovint convé conservar informació en fitxers. Aquest format de fitxer permetia especificar diferents espais de color pel JPEG, i el delmat d'alguns dels plans. Amb les càmeres digitals que directament conservaven les fotografies en JPEG i en fitxers, es va creure que la informació que permetia conservar el JFIF no era suficient. Va aparèixer l'EXIF, que permet conservar en un fitxer molta més informació de les circumstàncies de la fotografia (estat de la càmera, sensors, etc.), a més de la codificació JPEG. Molta gent sol anomenar els contenidors JFIF i EXIF indistintament "Fitxers JPEG", o bé amb la reducció a tres lletres obligada sota sistemes MS-DOS, "Fitxers JPG".

Dins l'estàndard s'hi inclouen dues formes diferents de codificar la imatge: amb pèrdues (*lossy*) o sense (*lossless*). Aquesta última no ha esdevingut popular, i segurament qualsevol imatge JPEG que trobem haurà estat codificada amb pèrdues. D'ara en endavant parlarem exclusivament de la compressió amb pèrdues.

La imatge original, prèvia a codificar, la trobem definida amb un mapa de bits. Això significa que tenim la imatge dividida en espai, amb una malla

¹<http://www.jpeg.org/jpeg/index.html>

quadriculada de punts de color. Cada punt (píxel, píxel, *pel*, de l'anglès *picture element*) ens defineix el color en aquell lloc de la quadrícula. Si tenim una quadrícula de punts suficientment densa, a ull nu no podrem distingir-la i el conjunt semblarà una imatge de color contínua o de to continu – sobretot en el cas de que la imatge sigui d'origen fotogràfic, en comptes de generada artificialment.

La codificació JPEG, mitjançant les pèrdues i diversos sistemes de compressió de dades, permet convertir el mapa de bits a una representació més densa, tot i que no fidel, de la original. Si tenim la imatge codificada amb els plans de color Vermell-Verd-Blau (RGB), amb 8 bits per punt i component, tenim una densitat d'informació de 24 bits per punt (bpp). Amb la compressió JPEG podem arribar a relacions de 1,5 bpp sense escatimar gaire en qualitat d'imatge. En el cas de tenir el mapa de bits a partir d'una fotografia, almenys a una densitat de píxels de 72 ppi (72 píxels per polzada) en horitzontal i vertical, el nostre sistema de visió en prou feines pot distingir la pèrdua de definició del JPEG a 1,5 bpp.

En el procés de codificació (o compressió) JPEG, es llença part de la informació del mapa de bits original. Això vol dir que en el procés de descodificació tindrem només part de la informació original, i que per construir una nova imatge sencera, el descodificador s'ha d'inventar la part que falta. A les implementacions populars de descodificadors JPEG que hem vist [8], la invenció d'informació és molt simple, i no sempre encertada, considerant que cal obtenir una imatge el més semblant possible a l'original d'acord amb els criteris del nostre sistema visual. En aquest treball analitzarem tècniques que pretenen aprofitar millor la informació del JPEG per a reconstruir una versió més agradable de la imatge original codificada.

1.2 Popularitat

Des de l'establiment de l'estàndard JPEG com a norma ISO (10918-1) el 1994, i de la donació al domini públic de la implementació coneguda amb el nom de `libjpeg` [8], en versió 5 el mateix any, aquesta codificació ha esdevingut molt popular. Als inicis d'Internet, JPEG oferia la major densitat d'informació per a transmetre fotografies. La publicació de l'estàndard va alliberar l'ús de qualsevol restricció per patents, i la velocitat de procés de la `libjpeg` va permetre introduir suport pel format en moltes aplicacions dedicades al processament d'imatges, fins i tot en els ordinadors personals.

L'alta densitat de la informació JPEG, junt amb les restriccions d'ample de banda dels primers anys de l'Internet popular, van fer que la gent publiqués les seves fotografies en aquest format. Els límits en espai d'emma-

gatzematge també van causar que la gent conservés les fotografies només en aquest format. Aquesta corrent continua fins ara. Tot i que els amples de banda i l'espai per emmagatzemar informació han augmentat, la quantitat de fotografies intercanviades i conservades també ha augmentat. Amb l'arribada dels escàners i les càmeres digitals a l'ús domèstic, el públic adquireix cada vegada més imatges, i més grans (més punts al mapa de bits). Des de fa temps hi ha implementacions de la codificació JPEG en maquinari, i per exemple la majoria de càmeres digitals donen els seus resultats en format JPEG. Fins i tot, per desconeixença, prou gent utilitza aquest format per a guardar imatges artificials, que no són de to continu, simplement pel costum de veure les sigles JPEG arreu associades a imatges i a alta densitat d'informació.

El comitè de JPEG l'any 2000 va crear un nou format d'imatge que pretén oferir millor densitat d'informació que el JPEG, però malgrat que han passat set anys des de la seva definició, no ha guanyat prou popularitat com per substituir el JPEG de 1992. Parlant exclusivament de la codificació d'imatges de to continu, el comitè garanteix que el JPEG 2000 no està subjecte a patents, però la seva implementació lliure més popular, *jasper*², conté molt de càlcul i ús de memòria, fins i tot pels ordinadors personals actuals. Tampoc hi ha implementacions en maquinari per part dels fabricants de càmeres digitals. Tanmateix, per posar un exemple popular, el programari Adobe Acrobat® des de la versió 6 suporta la codificació JPEG2000 dins el seu format PDF 1.5. La majoria d'aplicacions d'Adobe, en la seva versió actual, suporten el format. D'acord amb alguns experts en el camp de fotografia, per això, per a densitats d'informació similars, opinen que el JPEG original encara els dóna una versió de la imatge més agradable, especialment en zones de textures d'alta freqüència.

Així doncs, ara per ara trobem arreu imatges codificades en JPEG. És probable que d'una fotografia digital que ens interessa, només tinguem accés a la seva codificació JPEG. Això és el que ha motivat aquest treball, i per això perseguim la millor descodificació JPEG possible. Ja des dels inicis de l'ús del JPEG que s'han proposat mètodes per a descodificar millor les imatges; mètodes complexes computacionalment, però que ofereixen resultats millors que la descodificació convencional. De tota manera, pràcticament cap d'aquests mètodes s'ha popularitzat, i en prou feines n'existeix alguna aplicació més enllà de l'article publicat en una revista científica. En aquest treball hem procurat donar una visió global sobre l'estat de l'art en el camp de la descodificació JPEG, així com implementar uns quants mètodes convinents, i fer-los públics en forma d'una aplicació informàtica. Amb això pretenem fer

²<http://www.ece.uvic.ca/~mdadams/jasper/>

disponibles els millors resultats d'anys de recerca acadèmica als nombrosos usuaris de JPEG. Si bé aquesta aplicació ha estat pensada per a versats en el camp de la imatge, hem procurat que l'usuari no necessiti coneixements experts de JPEG o de programació per a dur a terme les descodificacions.

1.3 Les pèrdues al JPEG

El procés de codificació segueix els següents passos, que més endavant expliquem amb més detall:

1. Dividim la imatge en plans de color.
2. Delmem alguns plans, si d'ells acceptem menys detall. Aquí tenim la primera pèrdua d'informació important.
3. Dividim cada pla resultant en blocs de 8x8 píxels.
4. Fem la transformada DCT (Apèndix A) de dues dimensions a cada bloc, i així n'obtenim la seva representació freqüencial en 64 coeficients.
5. Quantitzem els coeficients. Aquí hi ha la segona pèrdua d'informació important.
6. Codifiquem els coeficients quantitzats mitjançant tècniques sense pèrdues, i els guardem en un sol paquet de dades. Aquest serà el resultat de la codificació JPEG.
7. Col·loquem el paquet de dades en un fitxer a part (mitjançant JIF, JFIF, EXIF, etc.), o bé dins el contenidor que creiem convenient (Postscript, PDF, DJVU, etc.).

La descodificació, anàlogament, podem dividir-la en aquests passos:

1. Obtenim el paquet de dades JPEG del seu contenidor.
2. Recuperem els valors dels coeficients quantitzats del paquet (que havíem guardat sense pèrdues).
3. Desquantitzem els coeficients, decidint-ne un valor concret dins de l'interval de quantització corresponent. (Aquí el descodificador ha de prendre decisions, inventar, el millor possible)
4. Fem la transformada inversa DCT dels coeficients obtinguts, per arribar a la representació en blocs de mapa de bits de 8x8 píxels.
5. Els blocs constitueixen la informació espacial dels plans, i cal interpolar els plans delmats a la seva dimensió original. (Aquí el descodificador també ha d'inventar els nous punts al escalar la imatge)
6. Ja tenim una versió dels tres plans que determinaven la imatge, i podem entregar-la a qui l'hagi demanada.

1.3.1 Plans de color i delmat

Les nostres pantalles solen utilitzar tres plans per a determinar la imatge a mostrar en color: vermell, verd i blau. No generen qualsevol color de l'espectre, sino només punts d'aquests tres colors, cadascun amb intensitat variable. Les freqüències de llum dels tres colors s'avenen amb les de màxima sensibilitat dels nostres tres cons més importants, de manera que la pantalla produeix, pràcticament, l'excitació independent dels nostres cons, donant la sensació d'un espai de color continu ([7]). Un color groc, per exemple, degut a les respostes freqüencials dels cons, excita parcialment el con vermell i el con verd. La pantalla enganya els cons, excitant concretament una mica els del vermell, i també una mica els del verd. Els colors de blanc a negre es produeixen amb igual intensitat de llum vermella, verda i blava.

En el cas d'imatges en blanc i negre, d'un únic pla, l'etapa dels plans de color no participa en la codificació. En canvi, les imatges en color consisteixen en múltiples plans, com els de les pantalles, que les següents etapes tractaran de manera independent. El JPEG consisteix en una compressió d'imatges amb pèrdues, que introduirà defectes a les imatges que codifiquem. Volem que el sistema nerviós humà distingeixi poc aquests defectes, i veurem com una reorganització de la informació dels plans ens pot ajudar a aquest efecte. El nostre sistema nerviós distingeix variacions d'intensitat de llum millor que variacions de color, i en els plans RGB cada un porta una mica d'informació de la intensitat de llum. Hauríem d'introduir pocs defectes a cada pla, per a que el cervell en prou feines els distingís. De tota manera, hi ha una transformació de color que ens ajuda a distribuir la informació de la imatge en plans més útils. Els plans RGB es poden convertir mitjançant unes regles matemàtiques a tres nous plans que defineixen igualment la imatge: intensitat de llum, blavor i vermellor (YCbCr). Així tenim la informació separada en plans que el nostre cervell considera de manera diferent. Ara podem llençar molta informació dels plans de color (Cb i Cr), més que no pas podríem llençar dels plans RGB originals, si ho comparem en termes de percepció visual.

Quan parlem d'imatges amb formes reconeixedores, aquestes formes són molt importants a l'hora d'avaluar la percepció de la imatge. Sovint les formes venen molt determinades per la lluminositat general (pla Y), i segurament per això l'ull humà es comporta més sensible als defectes del pla Y que als dels plans Cb i Cr ([7]). En la majoria de codificadors JPEG veurem que els plans de color Cb i Cr es delmen i les següents etapes els quantitzen a menys qualitat. La majoria de càmeres digitals delmen per maquinari a 2:1 només en horitzontal, mentre que molts codificadors per programari delmen a 2:1 tant en horitzontal com en vertical.

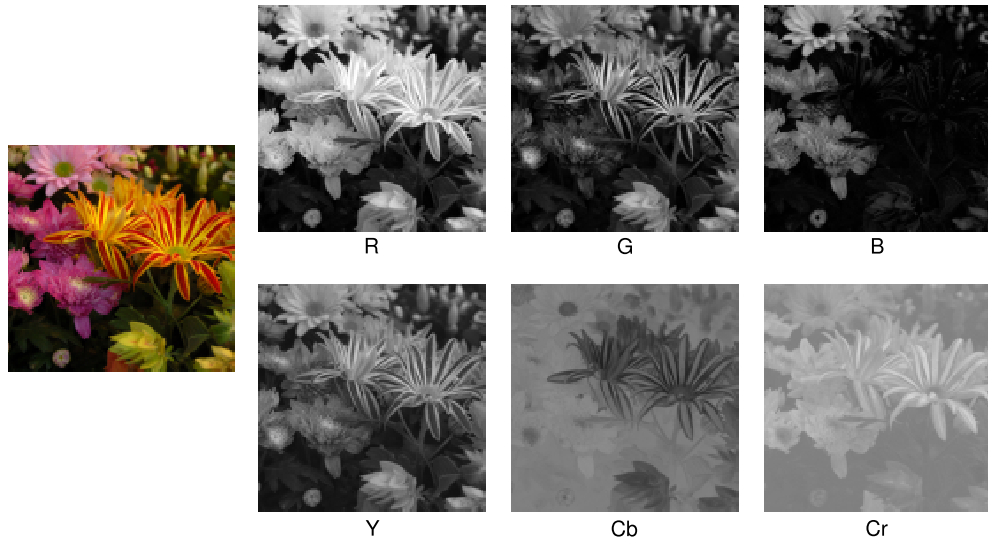


Figura 1.1: Plans RGB i YCbCr d'una imatge en color.

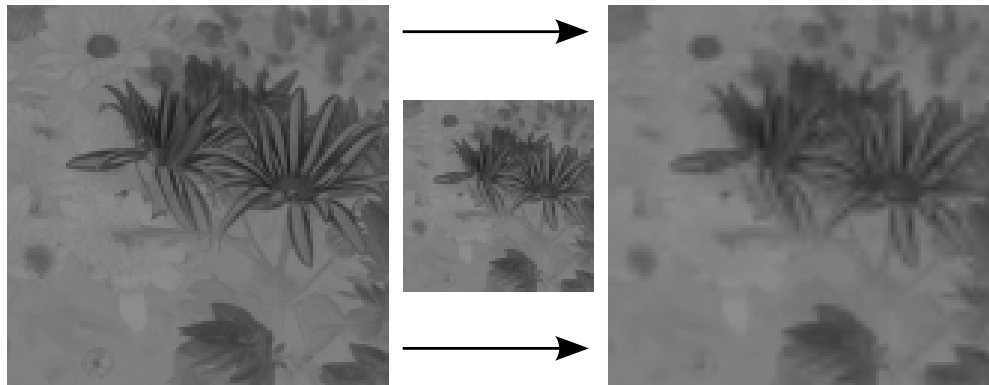


Figura 1.2: Delmat 2:1 en horitzontal i vertical del pla Cb, i posterior interpolació lineal de la versió delmada.

A la descodificació, cal interpolar els plans delmats a la seva mida original, mitjançant interpolacions o altres tècniques. Convencionalment (per defecte a la `libjpeg` [8]) es fa una interpolació lineal. I per a suplir la imatge final a la pantalla, cal invertir el canvi de plans de YCbCr a RGB.

El delmat dels plans de color provoca una incomoditat visual sovint detectable a les imatges JPEG descodificades: el *vessament de color* (*color bleeding*). A la Figura 1.3 podem veure'l.



Figura 1.3: A l'esquerra, l'original. Al mig, JPEG amb màxima qualitat al pla de lluminositat, i qualitat 40 de `libjpeg` als plans de color, amb delmat 2x2:1x1:1x1 (als dos plans de color la meitat de punts en horitzontal i vertical que al pla de lluminositat). Es pot apreciar el vessament de color sobre les finestres. A la dreta, JPEG amb la mateixa codificació per plans, però sense delmar els plans de color. Ja no s'hi aprecia tant el vessament.

4:4:4	1x1,1x1,1x1	Igual resolució de lluminositat i color
4:2:2	2x1,1x1,1x1	La meitat de punts de color en horitzontal
4:2:0	2x2,1x1,1x1	La meitat de punts de color en horitzontal, i la meitat en vertical
4:1:1	4x1,1x1,1x1	Un quart de punts de color en horitzontal

Taula 1.1: Notacions pels delmats dels canals de color per a les components YCbCr. La segona és típica de les càmeres digitals, i la tercera, el delmat per defecte al guardar imatges a diversos programes informàtics.

1.3.2 Blocs i quantització dels coeficients

Cada pla de la imatge es parteix en blocs de 8x8 píxels, i cada bloc es codifica de manera independent. La informació es llença per bloc, independentment de la informació dels blocs veïns. Degut a això, sovint podem distingir molèsties visuals precisament a les vores entre blocs, on la informació pertany a dos blocs que el codificador ha considerat independentment.

Dels 64 punts d'intensitat de color que determinen cada bloc de 8x8 píxels obtenim 64 coeficients mitjançant la transformada DCT 2D (Apèndix A). Aquests es solen ordenar en freqüències de menor a major, utilitzant l'ordre de Zig Zag (Figura 1.4) definit a l'estàndard de JPEG.

Les taules de quantització determinen la quantitat d'informació que es llença per cada un dels 64 coeficients. Es sol utilitzar una taula de quantització diferent per cada pla del JPEG. S'han dut a terme nombrosos experiments per a decidir quins coeficients són més importants per a degradar mínimament la imatge, establint un grau d'importància entre ells. En la majoria d'aplicacions que codifiquen imatges en JPEG l'usuari té l'opció d'escollir la densitat de la informació codificada, que consisteix en quantitzar els coeficients més o menys, mantenint el grau d'importància entre ells. No tots els codificadors utilitzen els mateixos patrons de taules de quantització. Les càmeres digitals tenen limitacions als algorismes en maquinari, els programes de manipulació fotogràfica professionals poden tenir els seus propis estudis de les taules de coeficients, i la `libjpeg` [8] simplement segueix les taules recomanades a l'estàndard ([10]). Podem veure un exemple de quantitzadors a la Taula 1.2.

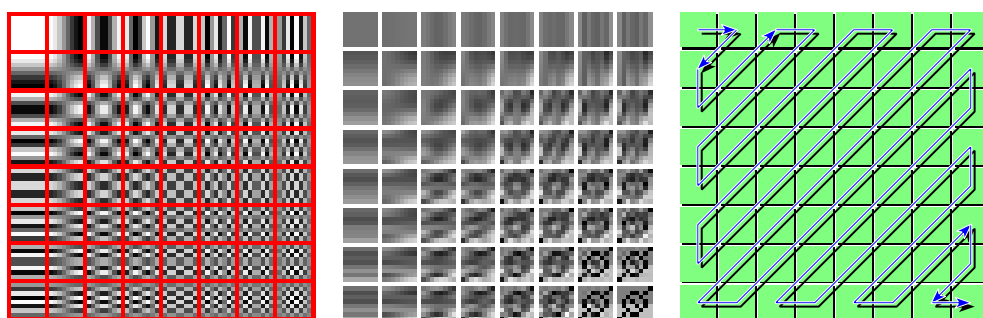


Figura 1.4: A l'esquerra, funció base de cada coeficient de la DCT 2D. Al mig, contribució parcial dels coeficients de la DCT; cada bloc s'ha reconstruït considerant nuls els coeficients de més altes freqüències. A la dreta, la serialització dels coeficients en Zig Zag ([10], 4.3). Les imatges primera i tercera les hem obtingut de la Viquipèdia [32].

La quantització dels coeficients consisteix en conservar només el valor enter resultat d'arrodonir la divisió pel quantitzador. El primer coeficient (la component contínua DC) no es quantitza independentment com els altres, sinó que es quantitza la diferència amb el primer coeficient del bloc anterior. Per cada un dels 63 coeficients AC provinents de la transformada DCT 2D (y_i , on $i = 2, 3, \dots, 64$) i per la diferència amb la component DC anterior (y_1)

80	55	50	80	120	200	255	305
60	60	70	95	130	290	300	275
70	65	80	120	200	285	345	280
70	85	110	145	255	435	400	310
90	110	185	280	340	545	515	385
120	175	275	320	405	520	565	460
245	320	390	435	515	605	600	505
360	460	475	490	560	500	515	495

Taula 1.2: Taula de quantització pel pla d'intensitat de llum segons la `libjpeg` [8] a qualitat 30, Q_i , $i = 1, 2, \dots, 64$ segons l'ordre de Zig Zag.

obtenim un enter arrodonint:

$$\hat{y}_i = \left\lfloor \frac{y_i}{Q_i} \right\rfloor, \quad i = 1, 2, \dots, 64$$

La recuperació convencional de cada coeficient, especificada a l'estàndard [10], consisteix en multiplicar pel quantitzador, $z_i = \hat{y}_i Q_i$. Podem veure un exemple dels coeficients d'un bloc a la Figura 1.5. De tota manera, a la descodificació, del coeficient original només sabem que es troba dins l'*interval de quantització*:

$$y_i \in Q_i [\hat{y}_i - 0,5, \hat{y}_i + 0,5] \quad (1.1)$$

A la pèrdua d'informació deguda a la quantització dels coeficients, un cop se'ls ha aplicat la transformada DCT 2D inversa, se l'anomena *soroll de quantització*. Té diverses conseqüències visibles a la imatge descodificada de manera convencional i en podem veure una classificació a la Figura 1.6.

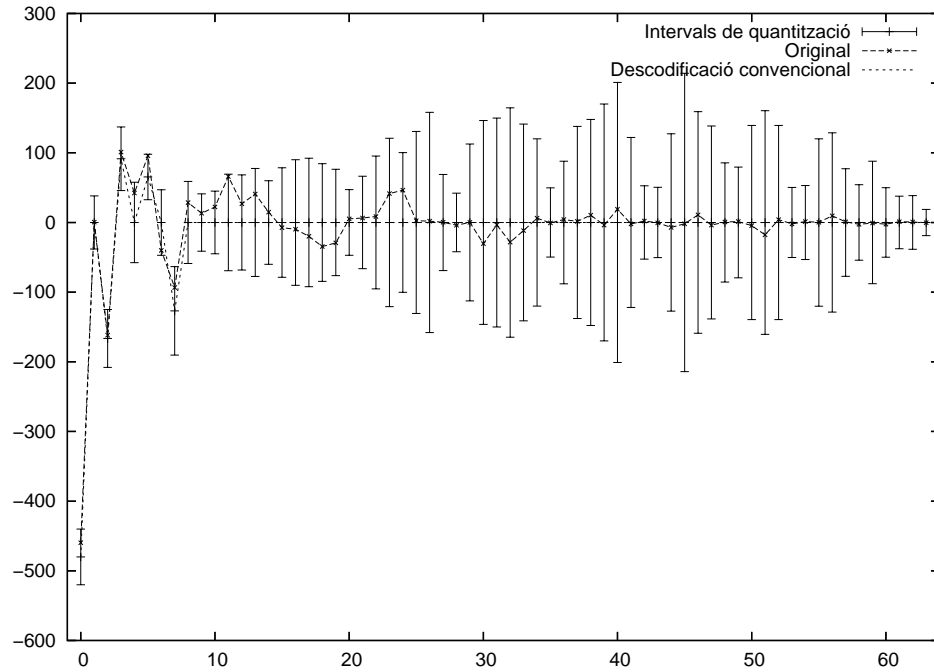


Figura 1.5: Coeficients del bloc situat a la fila 17, columna 17, del pla de lluminositat de la Lena de 256x256, amb una compressió a qualitat 30 segons la `libjpeg` [8]. Estan ordenats de menor a major freqüència, com marca l'ordre de Zig Zag de JPEG. Es poden apreciar els intervals de quantització, i la pèrdua d'informació sobretot a les freqüències més altes.

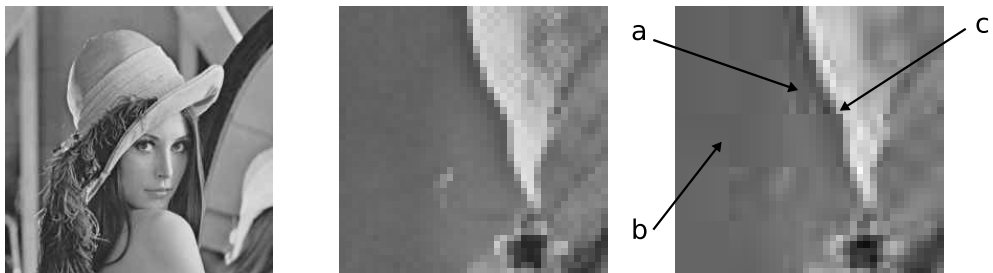


Figura 1.6: A l'esquerra, pla de lluminositat de la Lena de 256x256 píxels, a qualitat 40 segons la `libjpeg`. A la dreta, tall on podem veure els efectes *a)* d'ones (*ringing*) als blocs que tenen un canvi brusc d'intensitat pel mig, *b)* de blocs (*blocking*), entre blocs a les superfícies de lluminositat similar, i *c)* d'escala (*staircase*), quan apareixen graons a una zona de canvi d'intensitat. Al mig, el mateix tall de la imatge original, per comparar.

Capítol 2

Qualitat de les imatges

En aquest treball ens proposem millorar la descompressió d'imatges JPEG. Per tant, cal que considerem quantificar la qualitat de la imatge resultant.

Una mesura molt utilitzada, simplement pel fet de ser purament de teoria del senyal, és l'*error quadràtic mitjà* (MSE - *Mean Squared Error*, [22]). Aquesta mesura ens permet comparar imatges descodificades amb la versió original. Per cada pla de la imatge que ens interessi, considerem error la diferència entre cada punt de la imatge original i el punt equivalent de la versió descodificada. Així, per una imatge original de M columnes i N files,

$$MSE = E[(x - \hat{x})^2] = \frac{1}{MN} \sum_{m=1}^M \sum_{n=1}^N (x_{m,n} - \hat{x}_{m,n})^2$$

on $x_{m,n}$ representa la intensitat d'un punt de la imatge original, i $\hat{x}_{m,n}$ l'anàleg de la versió descodificada.

Als articles que fan referència a comparació d'imatges amb l'objectiu de mesurar la similitud encara es fa servir més una funció de l'MSE, la *relació senyal soroll de pic* (PSNR - *Peak Signal-to-Noise Ratio*), que es sol expressar en decibels i es defineix així:

$$PSNR = 10 \log_{10} \frac{255^2}{MSE}$$

A molts articles on proposen mètodes per millorar la descompressió JPEG justifiquen les millores amb un augment de la PSNR respecte a la descodificació convencional. A la pràctica, la PSNR es considera una mesura adequada quan es comparen imatges codificades a alta qualitat. En general, la PSNR mesura bé l'error percebut quan no hi ha gaires diferències a zones d'altres freqüències. Concretament, nosaltres no distingim gaire els desfasaments a altes freqüències, i en canvi la PSNR és molt sensible a desfasaments. En podem veure un exemple a la Figura 2.1.

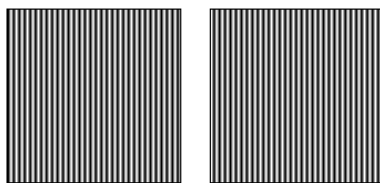


Figura 2.1: Aquestes dues imatges artificials les veiem molt semblants, però una està en contrafase amb l'altra. En aquest cas, la mesura de PSNR ens donaria un valor molt baix.

Per altra banda, la mesura de PSNR no es basa en peculiaritats del nostre sistema de visió, i al cap i a la fi és aquest qui jutjarà les imatges. Tot i que els judicis depenen de cada individu observador, sí que hi ha uns patrons comuns al sistema de visió humà (HVS – *Human Vision System*). Com que volem una mesura calculable de la qualitat de la imatge, podem aprofitar els resultats de diverses investigacions que s'han dut a terme sobre aquest tema. Hi ha un procediment comú a tots els que hem analitzat: trobar un model matemàtic que doni resultats correlats a mitjanes d'opinions d'individus (MOS – *Mean Opinion Score*).

La literatura de processat d'imatge ens proposa molts mètodes per a determinar la fidelitat d'una imatge recodificada. A [29] podem trobar una bona explicació de perquè la PSNR encara s'utilitza tant en mesures de qualitat, i una discussió sobre perquè la PSNR no és adequada per a mesurar la *fidelitat del senyal visual*.

A [25] fan una bona revisió de mesures referencials (FR – *Full-reference*, que comparen imatges distorsionades amb la seva original) existents, i a més proposen un nou mètode de mesura amb qualitats interessants sobre la PSNR: el VIF (*Visual Information Measure*). Aquest es basa en jutjar elements rellevants, considerant estadístiques d'escenes naturals, mitjançant transformades Wavelet dels plans i l'anàlisi de les subbandes a la imatge de referència i la distorsionada.

Per altra banda, el nostre sistema de visió pot distingir la influència d'una distorsió sovint sense haver de comparar amb la imatge original. En el cas concret de la descodificació JPEG, hi ha diversos mètodes No-Referencials (NR) proposats. Aquí en citem tres dels més utilitzats als articles que hem analitzat:

- *MSDS – Mean Squared Difference of Slope*: proposat a [18], aquest mètode pretén mesurar l'efecte de blocs. Es basa en analitzar les pendents d'intensitat a les zones entre blocs i a la frontera dels blocs adjacents. A imatges JPEG d'alta qualitat, el pendent d'intensitat abans del límit de bloc, i el d'entre blocs, serà similar. En canvi als JPEG

de poca qualitat, el pendent d'intensitat abans del canvi de bloc serà diferent al que hi ha entre blocs. Aquestes diferències es sumen, tant les mesurades en horitzontal com en vertical i en diagonal, per a donar un únic valor NR.

- *GBIM – Generalized Block-edge Impairment Metric*: a [33] proposen mesurar les diferències d'intensitat dels punts que estan al límit dels blocs consecutius, i comparar-los amb les diferències d'intensitat de punts que no estan a límits de blocs. Comparteix objectiu amb l'MSDS, mesurar l'efecte de blocs, considerant que els canvis bruscs d'intensitat de la fotografia original en general no coincidiran amb els límits blocs.
- *jpeg_score*: a [30] proposen una mesura similar al *GBIM*, però que a més té en compte que la imatge ha de tenir *detall* (altes freqüències). Així doncs, contempla els canvis d'intensitat als punts entre blocs B , els canvis d'intensitat als punts interiors dels blocs A , i els canvis de sentit de la derivada de la intensitat Z , independentment dels punts que tinguin la imatge. Llavors, proposen la mesura S amb la següent forma:

$$S = \alpha + \beta B^{\gamma_1} A^{\gamma_2} Z^{\gamma_3}$$

on els paràmetres $\alpha = -245,9$, $\beta = 261,9$, $\gamma_1 = -0.0240$, $\gamma_2 = 0.0160$ i $\gamma_3 = 0.0064$ són estimats a partir d'una correlació entre S i MOS sobre una estadística de 30 imatges originals, 90 versions comprimides i 53 individus. Les MOS estaven acotades entre 1 i 10, llavors en la majoria dels casos $1 < S < 10$.

El programari vinculat a aquest informe, tot i que ofereix algunes de les mesures matemàtiques presentades (*GBIM* i *PSNR*), fonamentalment permet comparar còmodament imatges a ull nu. Hem vist que hi ha moltes mesures per avaluar les restauracions d'imatges JPEG, i per això no hem dedicat gaire temps a implementar-les, prioritzant que l'usuari pugui avaluar les imatges segons el seu criteri, que segurament serà el més adequat a les seves circumstàncies.

Capítol 3

Tècniques

a A l'apartat 1.3 hem explicat quines pèrdues de senyal trobem al JPEG, i quins són els marges de llibertat que tenim a l'hora d'intentar recuperar la imatge original. Des de l'aparició de l'estàndard JPEG que s'han proposat mètodes per a millorar la descodificació mitjançant algorismes més complexos que el convencional. Hem procurat estudiar l'estat de l'art del JPEG, i classificar els mètodes més interessants segons les tècniques en què es basen.

3.1 Projectió sobre l'espai de quantització

Aquesta tècnica no millora les imatges en si, però l'expliquem a part perquè forma part de molts dels mètodes analitzats. Podem diferenciar els mètodes en dos grans grups: els de *dins la cadena de descodificació*, i els de *post-processat*. Els primers consideren tota la informació del JPEG, i els segons treballen exclusivament a partir de la imatge obtinguda després d'una descodificació convencional. Els primers tenen un avantatge clar sobre els altres, ja que poden accedir als intervals de quantització, als plans abans d'interpolar-los i a la codificació del color.

Considerem la descodificació d'un pla del JPEG, no necessàriament amb el mètode convencional. Si al recodificar el pla amb les mateixes taules de quantització obtenim la mateixa versió codificada que de l'original, vol dir que hem aplicat una descodificació *no destructiva*. Els coeficients que hem restaurat podrien ben ser de la imatge original, ja que són coherents amb la informació del JPEG. Ara bé, com podem veure a la Figura 3.1 el marge pot ser bastant gran segons les taules de quantització. Els mètodes de *post-processat* no poden garantir aquesta coherència amb el JPEG ja que desconeixen les taules de quantització. Per exemple, amb programes informàtics de retoc fotogràfic es sol utilitzar el filtrat Gaussià selectiu per a eliminar

soroll, entre d'altres, de la descodificació JPEG convencional.



Figura 3.1: *a)* Imatge Lena codificada a qualitat 40 amb la `libjpeg`, *b)* Descodificació segons els valors absoluts més baixos dins dels intervals de quantització, *c)* Descodificació segons els valors absoluts més alts.

Alguns mètodes de descodificació basen la seva estimació d'imatge en filtres espacials, a vegades fins i tot en processos iteratius. Per assegurar-se que la seva imatge no divergeix de la informació del JPEG, duen a terme el que anomenem *projeccions sobre l'espai de quantització*. Quan tenen el nou pla espacial estimat, li fan la transformada DCT per blocs de 8x8, com a la codificació JPEG. Els coeficients obtinguts z_i els comparen amb els intervals de quantització (eq. 1.1), i si se'n surten, els consideren al límit de l'interval.

$$z'_i = \begin{cases} (\hat{y}_i - 0.5)Q_i & \text{si } z_i < (\hat{y}_i - 0.5)Q_i \\ (\hat{y}_i + 0.5)Q_i & \text{si } z_i > (\hat{y}_i + 0.5)Q_i \\ z_i & \text{altrament} \end{cases} \quad (3.1)$$

Aquest mètode també es coneix a la literatura amb el nom de *projection to QCS* (Quantization Constraint Set) o *projection to the constraint space*, i s'utilitza als mètodes [2], [20], [23], i en general a qualsevol mètode iteratiu dels que hem vist.

Alguns articles proposen projectar sobre NQCS (*Narrow Quantization Constraint Set*), que consisteix en agafar un marge més estret que el de $[-0.5, 0.5]$ al voltant del centre de l'interval de quantització. A [16], per exemple, consideren millors els resultats amb un marge de $[-0.3, 0.3]$.

3.2 Coeficients com a variables aleatòries

La descodificació convencional d'agafar el punt mig de l'interval de quantització per cada coeficient compleix la *no destrucció*. Aquest cas resulta de

considerar l'interval com una variable aleatòria uniforme, on el punt mig és el millor estimador de la variable.

Alguns articles suggereixen altres distribucions de probabilitat. Per exemple, a [27] proposen utilitzar una v.a. exponencial, de mitjana a determinar segons la imatge (veieu la Figura 3.2). A [23], [28] i a [5] mencionen utilitzar una v.a. Laplaciana. El primer, basant-la amb la imatge a descodificar. El segon, basant-la amb estadístiques d'altres imatges.



Figura 3.2: *a)* Imatge Lena original, *b)* Descodificació convencional de qualitat 40 segons la `libjpeg`, considerant els intervals com v.a. uniformes, *c)* descodificació de la mateixa imatge considerant v.a. exponencials de mitjana $\mu = 1$.

3.3 Segmentació i suavitzat

Un dels efectes que més es distingeixen a les imatges codificades amb una relació de compressió alta (més denses) és el de blocs. A les zones de baixes freqüències, on en prou feines s'haurien de distingir canvis de lluminositat, apareixen discontinuïtats segons el patró de blocs del JPEG. El cas més extrem el trobem quan a zones amb freqüències baixes es conserva només el coeficient de component contínua (el primer, en ordre Zig Zag).

Aquest tipus d'efecte de blocs es pot arreglar amb un simple suavitzat, per exemple amb un filtre de Gauss. Així, el que fan alguns algorismes consisteix en distingir les zones de baixes freqüències, i aplicar-los el suavitzat. A [3], [1], [21], [27], utilitzen filtres 2D simples; a [15], [12] i [5] fan servir filtres adaptatius 2D; a [6] i a [16] fan servir filtres 1D; i a [17] fan servir interpolació lineal sobre la component contínua.

Alguns algorismes més complexos suggereixen distingir i processar també les zones d'altres freqüències. A [3] i [17] fan servir un filtre amb molt poc pes dels punts del voltant; i a [1], [21] i [27] fan servir filtres direccionals.

3.4 POCS - Projectió sobre conjunts convexos

Els mètodes basats en POCS solen donar molts bons resultats, i apareixen a la literatura des dels inicis de l'ús del JPEG. Malauradament l'algorisme és bastant més lent que altres, ja que consisteix en dur a terme passos iteratius.

Referint-me a l'explicació de POCS de [35], assumim que les imatges f , representades amb un vector, són elements d'un espai de Hilbert H . Llavors, per qualsevol $f \in H$, la seva projectió Pf en un conjunt tancat convex $C \subset H$ es defineix com l'element més proper a f dins C . Això és:

$$\|f - Pf\| = \min_{g \in C} \|f - g\|$$

Considerem que la projectió Pf ve determinada exclusivament per f i C ; o sigui, per a cada conjunt tancat convex, tenim el seu projector.

Si tenim diversos $C_i \subset H$, $i = 1, 2, \dots, m$, i igual nombre de projectors P_i sobre C_i , podem veure que per qualsevol imatge inicial f_0 podem obtenir una seqüència d'imatges $\{f_k\}$,

$$f_{k+1} = T_m T_{m-1} \cdots T_1 f_k$$

on $T_i = I + \lambda_i(P_i - I)$, $0 < \lambda_i < 2$, $i = 1, 2, \dots, m$. Aquesta seqüència convergeix al punt

$$f^* \in C_0 \triangleq \bigcap_{i=1}^m C_i.$$

Relacionant POCS amb la millora d'imatges, el que ens cal fer és tenir un conjunt tancat convex C_i (i el seu projector P_i) per cada característica de la imatge. Llavors, per a m característiques diferents, podem trobar la imatge f^* que millor les compleix. Iterativament, apliquem tots els projectors, partint de la primera aproximació f_0 a la imatge desitjada. Sovint aquesta primera aproximació consisteix de la descodificació convencional de JPEG.

Els mètodes basats en POCS solen tenir el QCS com a conjunt tancat convex involucrat C_1 , i solen presentar només un altre conjunt que els distingeix dels altres mètodes ($m = 2$). A [35] intenten reduir les discontinuïtats dels punts entre blocs amb C_2 i a [31] utilitzen un C_2 basat amb prèvia classificació dels punts en regions homogènies de la imatge. En canvi, [34] incorpora fins a sis conjunts per eliminar l'efecte de blocs i amb especial incís l'efecte d'ones.

3.5 Maximum A Posteriori

Aquesta tècnica és similar a la de POCS en els aspectes de que és iterativa, i té projeccions sobre conjunts convexes. Estrictament, a POCS els mètodes venen definits pels conjunts convexes involucrats. Amb la tècnica de MAP el que es busca és la maximització d'una funció determinada, considerant la informació que sabem de la imatge. Aquesta funció avalua les imatges segons diferents mesures preses de la imatge, i hauria de donar un valor alt per mesures que considerem d'imatge més correcta, i un valor baix per mesures que considerem d'imatge incorrecta. Per exemple, podem establir una mesura de si hi ha poques o moltes diferències entre el color dels píxels entre blocs. Com més diferències hi hagi entre aquests píxels, més baix és el valor que retorna la funció.

En el cas del JPEG, l'objectiu és maximitzar la funció sempre i quan mantenim els coeficients dins els intervals de quantització. Si \hat{z} és la nostra imatge estimada a la descodificació, Z és el conjunt d'imatges amb coeficients dins el QCS, i $P(z)$ una funció que mesura la imatge tal com hem explicat abans:

$$\hat{z} = \max_{z \in Z} P(z)$$

Els mètodes que hem vist utilitzen funcions $P(z)$ dues vegades derivables, i minimitzen mitjançant el *mètode del gradient*. A [20] i a [23] utilitzen com a funció un model de camp aleatori de Huber-Màrkov. A [2] utilitzen una versió adaptada de la norma de Variació Total (TV).

3.6 Altres mètodes de recuperació de plans

Alguns mètodes ([11], [14]) utilitzen transformacions d'ondetes, ja que aquestes aporten informació freqüencial i espacial. Amb la primera poden distingir zones de més i menys textures per aplicar *segmentació*, i amb la segona poden tractar els efectes de bloc, que estan determinats en espai.

A [19] i [24] utilitzen la recompressió JPEG de versions desplaçades de la imatge descodificada de manera convencional, fent una mitjana de les múltiples versions de les imatges desplaçades descodificades.

3.7 Recuperació del color basada en lluminositat

Tal com hem vist a l'apartat 1.3.1, el JPEG aïlla tota la informació de tonalitat de color en dos plans que es codifiquen a menys qualitat que el de

lluminositat. Totes les tècniques que hem presentat fins ara es poden utilitzar per recuperar qualsevol dels plans del JPEG, però a la literatura també trobem tècniques que utilitzen la correlació entre lluminositat i color per a recuperar millor els plans cromàtics, sovint delmats fins i tot.

A [4] detecten les cantonades del pla de lluminositat, i les utilitzen amb un algorisme adaptatiu per corregir els vessaments de color deguts a l'escalat. A [26] ponderen els canvis d'intensitat dels plans de color d'acord amb els canvis d'intensitat del pla de lluminositat, per a imatges YUV en general. Aquest l'últim l'hem adaptat al delmat del JPEG, explicat en detall a l'Apèndix B.2.

Capítol 4

Programari

Com hem vist a l'apartat anterior, trobem nombrosos articles sobre la millora de la descompressió JPEG. Malauradament, no trobem els algorismes implementats en cap aplicació popular. GIMP i Adobe Photoshop®, possiblement els processadors de mapes de bits més populars, només suporten la descodificació convencional. A la última versió (6b) de la `libjpeg` tampoc hi trobem res d'especial. A la seva documentació hi trobem una declaració d'intencions per incorporar mètodes de restauració a la versió 7, però sembla que el desenvolupament d'aquesta llibreria es va aturar precisament a la 6b el 1998.

Així doncs hem trobat oportú construir una aplicació informàtica que ofereixi als professionals de la imatge accés relativament fàcil als algorismes de restauració més destacats. Com explicàvem al primer apartat d'aquest document, la codificació JPEG es continua utilitzant activament, i per tant molts usuaris es poden beneficiar d'una iniciativa com aquesta.

Quan hem buscat implementacions dels algorismes de restauració analitzats, n'hem trobat molt poques, i la majoria d'aquestes poques estaven escrites en Matlab. Els intèrprets de Matlab els trobem pràcticament només a les universitats o dins departaments d'investigació d'empreses, però la majoria de professionals de la imatge ni els posseiran ni els sabran utilitzar. Nosaltres hem desenvolupat tota l'aplicació i els algorismes en C i C++, ja que ens semblen els llenguatges més populars i a l'abast de qualsevol gràcies a diversos compiladors i enllaçadors lliures disponibles. Aquí tenim un llistat de les principals característiques del programa:

- Només permet manipular una imatge JPEG alhora, junt amb versions restaurades o que teníem guardades a disc.
- Tot el procés i emmagatzematge intern de la informació de les imatges es realitza en coma flotant de 32 bits (`float` en C), per evitar masses

pèrdues per arrodoniment als càlculs.

- El programa requereix que l'usuari entengui la part de descodificació JPEG que hem explicat al primer apartat, ja que és ell qui determinarà com es realitzaran diversos passos d'aquesta descodificació.

L'aplicació inclou implementacions dels algorismes de restauració que ens han semblat més prometedors, i també permet comparar còmodament diferents versions d'una imatge, bé a ull nu, bé amb mesures matemàtiques. A l'hora d'escollir els algorismes, ens hem regit pels següents criteris:

- No ens importa la complexitat de l'algorisme en temps de computació o memòria. No tractarem vídeo, sinó imatges estàtiques. Alguns articles presenten novetats exclusivament en el camp de la restauració ràpida, tot i que no aconsegueixen recuperacions tan bones com alguns algorismes més lents.
- Hem considerat que els exemples d'imatge presentats al mateix article ofereixen una bona idea del millor resultat que l'algorisme permet assolir. Creiem que alguns articles presenten molt poca millora (valorant subjectivament a ull nu) com per a tenir-los en compte a la nostra implementació.
- L'algorisme ha de permetre millorar imatges en general. Alguns només milloren determinades zones de molt baixes freqüències, o fins i tot només serveixen per a dibuixos artificials en comptes de fotografies.
- L'efecte de l'algorisme s'ha de caracteritzar per pocs paràmetres, i aquests han de tenir una relació monòtona amb els seus efectes. En la majoria de casos l'usuari no entendrà l'algorisme, i en variarà intuïtivament els paràmetres. Si l'usuari no pot entendre l'efecte que té cada paràmetre, segurament refusarà manipular-ne aquests paràmetres, o fins i tot no el tindrà en compte a l'hora de restaurar.

Podem classificar els algorismes segons la seva funció. A la nostra aplicació distingim:

- Mesures no-referencials. Per exemple, GBIM.
- Mesures referencials (comparació). Per exemple, la PSNR.
- Elements de la cadena de descodificació.

A qualsevol imatge que hàgim carregat dins de l'aplicació li podem aplicar els algorismes de mesura, i obtindrem els resultats en un nou quadre flotant de text. Per a les mesures referencials haurem d'escollir les dues imatges a comparar, en comptes de només una.

4.1 La cadena de descodificació

Un cop l'usuari hagi determinat quina imatge JPEG vol recuperar, li caldrà decidir com es descodificarà. L'aplicació li donarà un esquema de la *cadena de descodificació* amb les baules buides, similar al de la Figura 4.1. Segons on es troben de la cadena, les baules fan referència a una part o altra del procés de descodificació. A cada tipus de baula li correspon un tipus d'algorisme. Considerem només aquests quatre tipus:

- Desquantització i transformada inversa de cada pla.
- Restaurador de pla
- Escalador dels plans
- Canvi de plans de color. Per exemple, de YCbCr a RGB.

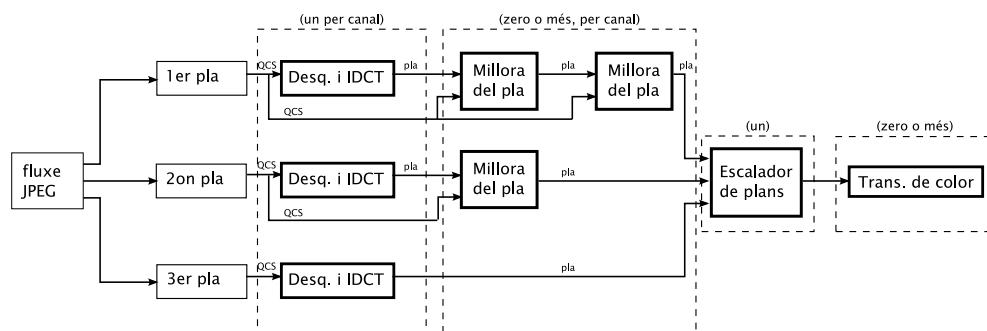


Figura 4.1: Cadena de descodificació d'un JPEG amb tres plans de color, amb les baules (amb el contorn més gruixut) on hi podem col·locar diversos algorismes. Les línies de punts separen els diferents tipus d'algorismes acceptats. D'esquerra a dreta: *desquantitzador i IDCT2D*, *restauradors de pla*, *escalador*, *canvis de plans de color*.

L'aplicació posarà a disposició de l'usuari uns quants algorismes de cada tipus, i aquest podrà col·locar els que cregui adients a la cadena, com si es tractessin de baules. Alguns dels algorismes que col·locarem a les baules són configurables. Això vol dir que el resultat que donin dependrà tant de l'entrada de l'algorisme (provinent de la baula anterior) com de la configuració dels seus paràmetres interns. Podem veure l'aspecte de l'aplicació mentre intentem restaurar un JPEG de poca qualitat de la imatge Lena a la Figura 4.2. Per entendre els noms d'algorismes que s'hi distingeixen, podem consultar la Taula 5.1, pàg. 37.

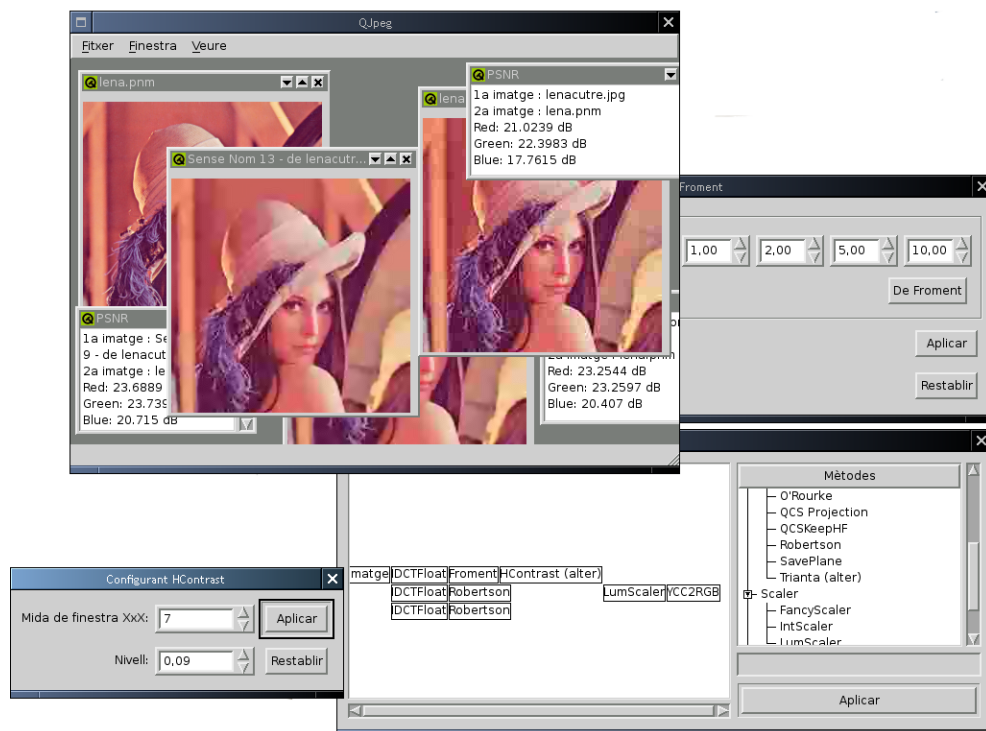


Figura 4.2: Mostra de les principals finestres del programa. La principal, amb versions de la imatge a restaurar. A sota, la cadena de descodificació. A dalt a la dreta, de fons, la configuració de la baula de Froment. A l'esquerra, configuració de la baula HContrast.

4.2 Desquantització i IDCT 2D de cada pla

entrada els intervals de quantització d'un pla de la imatge codificada

sortida una imatge d'intensitat espacial del pla

Aquest element té una doble funció: escollir quins punts de l'interval de quantització determinaran els coeficients freqüencials concrets, i dur a terme la transformada DCT 2D inversa.

A l'hora d'escriure aquest document, a l'aplicació hi ha dos algorismes disponibles d'aquest tipus. El primer coincideix amb la descodificació convencional de JPEG escollint el punt mig dels intervals. I el segon considera la informació dels coeficients AC (tots menys la component contínua DC, el primer coeficient de cada bloc) com variables aleatòries exponencials d'igual mitjana, com a [27]. L'usuari pot configurar-ne la mitjana.

4.3 Restaurador de pla

entrada els intervals de quantització d'un pla de la imatge codificada

entrada una imatge d'intensitat espacial del pla (imatge de partida)

sortida una imatge d'intensitat espacial del pla (imatge millorada)

La majoria d'algorismes de restauració de JPEGs que hem trobat encaixen dins aquesta categoria. Partint d'un pla inicial (als articles sol ser una descodificació convencional) utilitzen diverses tècniques per obtenir una versió millor del mateix pla. A l'apartat 3 ja hem parlat d'aquest tipus d'algorismes.

Cal destacar que als articles utilitzen imatges d'escala de grisos ja que exclusivament intenten millorar un pla de la imatge. A la nostra aplicació podem utilitzar aquests algorismes a qualsevol dels plans del JPEG. Igualment, podem encadenar tants algorismes d'aquests tipus com vulguem, on cada un determinarà la imatge inicial del seu successor.

Quan escrivim aquestes línies, l'aplicació dona implementacions de [20], [2], [23], [27], [19] i [24]. No tots aquests algorismes utilitzen la *projecció a l'espai de quantització* per assegurar el compliment de la informació del JPEG, així que també hem inclòs aquesta projecció com a algorisme independent. A part de les propostes d'articles, inclou un senzill algorisme de millora de les zones amb textura desenvolupat per nosaltres mateixos que aprofita la naturalesa del programa: combinar i encadenar tècniques. L'explicarem més endavant a l'Apèndix B.1. Finalment, la implementació actual

també inclou uns mòduls especials que ens permeten alterar els plans mitjançant una altra aplicació informàtica. Són els següents:

- *salvar pla a disc*. Si l'usuari col·loca aquesta funció dins d'una baula de restauració de pla, al aplicar la descodificació definida per la cadena, el pla d'entrada de la baula s'escriurà a disc. Llavors l'usuari pot utilitzar el seu programa preferit de manipulació fotogràfica aplicant les tècniques que cregui convenientes al pla, i guardar els canvis per a reincorporar-los a la cadena amb la funció que descrivim a continuació.
- *carregar pla de disc*. El pla d'entrada d'aquest algorisme s'ignorarà completament, i el pla de sortida vindrà determinat per un fitxer guardat a disc.

Aquestes dues funcions les hem utilitzat a la pràctica per millorar els plans amb un filtre del GIMP¹, el suavitzat Gaussià selectiu. Aquesta tècnica també la utilitzen combinada amb altres a [5] i [12], i alguns manuals de GIMP la recomanen per eliminar part del soroll d'algunes imatges JPEG. Si utilitzem les funcions de *salvar el pla*, apliquem el filtre amb el GIMP, *el carreguem a la cadena*, i després hi apliquem una *projecció sobre l'espai de quantització*, obtenim resultats competitius amb altres algorismes, segons el tipus d'imatge i el tipus de codificació JPEG.

4.4 Escalador dels plans

entrada n plans independents, possiblement alguns delmats

sortida n plans, tots de la mateixa mida

Tal com hem explicat a l'apartat 1.3.1, alguns dels plans de color de la imatge poden haver estat delmats abans de codificar-los independentment (DCT 2D i quantització). La imatge final descodificada ha de tenir tots els plans de color de la mateixa mida. Per això al descodificar ens cal interpol·lar dels punts dels plans delmats, de manera que obtinguem les seves dimensions originals. Cal recordar que el codificador delmarà els plans dividint les dimensions originals per un número enter, i els delmats més comuns són 4:2:2 i 4:2:0.

Al programa hem implementat tres tipus d'escaladors. El primer simplement duplica els punts per aconseguir les dimensions desitjades. Per exemple,

¹GNU Image Manipulation Program <http://www.gimp.org/>

si tenim els valors per punt en una fila 1,5,3,2,6, i ens cal doblar la seva mida horitzontal, l'algorisme ens donarà 1,1,5,5,3,3,2,2,6,6. La `libjpeg` també inclou una implementació d'aquest escalador.

El segon escalador, provinent també de la `libjpeg` (on l'anomenen *fancy upsampler*), interpola linealment els punts del pla final a partir de la versió delmada descodificada, tot i que només funciona pels delmats 4:2:2 i 4:2:0. En el cas de 4:2:2, interpola cada punt final amb els dos punts més propers del pla delmat. Hem de recordar que segons l'especificació JFIF [9], els plans delmats a la meitat en horitzontal, no tindran punts coincidents amb els del pla final. Així doncs, obtenim els punts de color per les posicions dels punts de lluminositat de la següent manera:

$$\hat{p}_o = \frac{3}{4}p_1 + \frac{1}{4}p_2$$

on \hat{p}_o és el valor d'un punt d'una fila final, p_1 és el del punt de la fila delmada més proper a \hat{p}_o , i p_2 el del segon punt més proper. La Figura 4.3 representa visualment el procés.

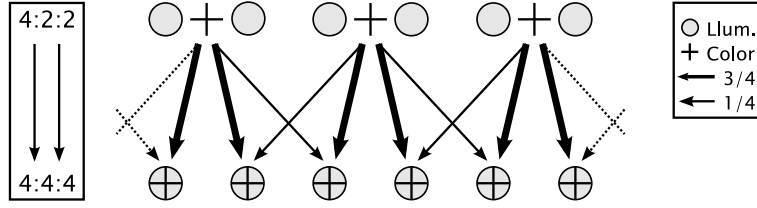


Figura 4.3: Reescalat d'una fila de la component de color de delmat 4:2:2 (a dalt) a la mida final determinada per la lluminositat (a baix), mitjançant interpolació dels dos punts més propers.

En el cas 4:2:0 el segon escalador interpola cada punt final mitjançant els quatre punts més propers:

$$\hat{p}_o = \frac{9}{16}p_1 + \frac{3}{16}(p_2 + p_3) + \frac{1}{16}p_4$$

on \hat{p}_o és el valor d'un punt del pla final, p_2 i p_3 són els dels altres dos punts més propers, i finalment p_4 el valor del punt menys proper dels quatre. Podem veure-ho gràficament a la Figura 4.4.

Finalment, hem implementat un tercer escalador, basat en [26], on utilitzem una tècnica similar a la interpolació lineal, però a més d'obtenir el color de cada punt final basant-nos amb els quatre punts de color més propers, ho ponderem segons les variacions de llum del pla de lluminositat (quan no ha estat delmat). A l'Apèndix B.2 expliquem l'algorisme en detall.

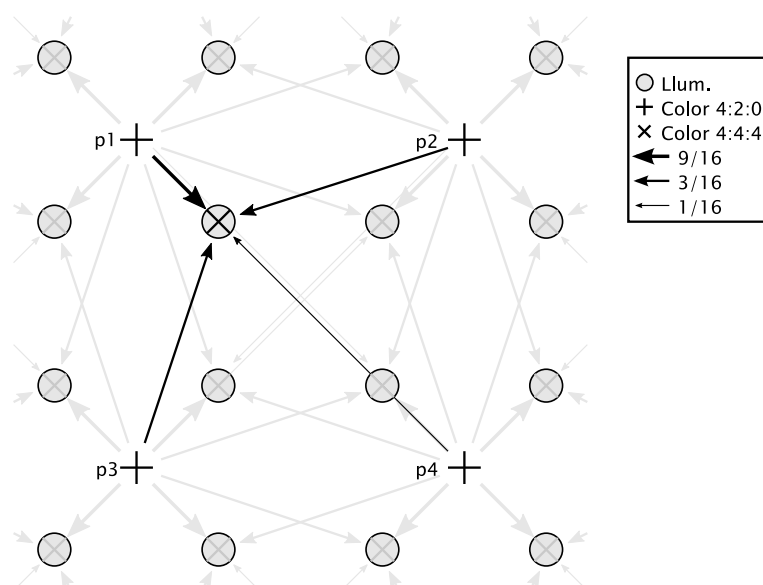


Figura 4.4: Reescalat d'una porció de la component de color de delmat 4:2:0 (en forma de creu dreta) a la mida final determinada per la lluminositat (en forma de cercles), mitjançant interpolació lineal entre els quatre punts més propers.

Tot i que fins ara hem parlat del cas general de que les imatges JPEG tenen unes mides múltiples de la mida de bloc (8 píxels), en realitat aquesta condició no s'exigeix a les imatges a codificar. La mida original de la imatge està escrita al contenidor del JPEG, i l'escalador del descodificador és l'encarregat de retallar els últims blocs en horitzontal i vertical per aconseguir la imatge final de la mida que toca. Així, si una imatge mesura 124 píxels d'amplada, s'haurà codificat amb 16 blocs de 8 columnes, però de l'últim bloc de cada fila (codificada d'esquerra a dreta) només tindrem en compte la meitat esquerra descodificada de 4 columnes d'amplada.

4.5 Canvi de plans de color

entrada $n > 1$ plans de color

sortida $n > 1$ plans de color

Aquests algorismes transformen els plans de color codificats del JPEG als plans exigits pel sistema gràfic del nostre ordinador. La nostra aplicació ara mateix necessita que les imatges finals tinguin *a)* un pla de lluminositat o *b)* tres plans vermell-verd-blau. Segurament la majoria dels nostres JPEG a

color estaran codificats amb lluminositat-blavor-vermellor, i per tant, per a mostrar-los correctament a l'aplicació, necessitarem un algorisme convertidor de YCbCr a RGB.

Les extensions JFIF permeten anotar en quins plans de color s'ha codificat la imatge. A la `libjpeg` per defecte es codifica amb YCbCr, però també s'hi poden indicar codificacions RGB o CMYK. Nosaltres només hem implementat el convertidor YCbCr a RGB, ja que no ens hem trobat imatges que utilitzessin altres configuracions de color.

Capítol 5

Anàlisi dels mètodes implementats

Al programa hem donat uns noms als algorismes que hem implementat. A la Taula 5.1 els teniu tots llistats, i permetran seguir millor els comentaris dels seus resultats.

IDCTFloat	Descodificació convencional per a un pla
IDCTFExp	Coefficients com v.a. exponencials, de [27]
Froment	[2]
O'Rourke	[20]
Robertson	[23]
Trianta	[27]
Nosratinia	[19]
HP	[24]
QCS Projection	Apartat 3.1
HContrast	Apèndix B.1
LoadPlane	Carregar el pla de disc
SavePlane	Salvar el pla a disc
IntScaler	Escalador que duplica el valor dels punts delmats
FancyScaler	Escalador interpolador lineal
LumScaler	[26], Apèndix B.2

Taula 5.1: Noms curts dels mètodes implementats al programa amb l'algorisme corresponent.

En aquest anàlisi no ens hem basat gaire amb mesures matemàtiques, ja que precisament un dels objectius que ens havíem proposat a l'hora de fer aquest programari consisteix en permetre a l'usuari comparar resultats a ull nu, variant paràmetres manualment. L'usuari dels algorismes, en aquest cas,

provarà d'aplicar diversos mètodes i a anar variant els seus paràmetres, fins a obtenir una imatge que l'acontenti pel seu objectiu. Els articles ofereixen mesures objectives dels resultats de diversos algorismes, i no sempre hi ha una correspondència clara entre aquestes mesures i l'opinió d'individuals. A l'avaluació hi prenen part almenys: el contingut de la imatge (persones, dibuixos, paisatges), la resolució (de quina distància es miraran els punts de la imatge), i l'objectiu de la restauració (impressió, postprocessat). Les mesures objectives que hem vist no tenen en consideració tots aquests paràmetres, i per tant no ens semblen suficients per avaluar prou útilment els resultats dels algorismes. En aquest apartat, quan comparem algorismes, ens referirem a les opinions d'uns pocs individus després d'haver escollit manualment tant els algorismes com els paràmetres de la cadena de descodificació.

5.1 Plans a molt poca qualitat

Aquest tipus d'imatges (qualitat < 40 a `libjpeg`) es caracteritzen per efectes de bloc molt importants i poc detall de textures. Partint d'una descodificació convencional, els algorismes iteratius no-adaptatius de *Froment*, *O'Rourke*, i *Robertson* són els que ens han donat millors resultats. El de *Trianta*, com que funciona de manera adaptativa segons informació espacial, no distingeix bé els blocs del JPEG de blocs de cantonades reals de la imatge. Els de *Nosratinia* i *HP* suavitzen excessivament la imatge, comparant amb els tres primers que hem mencionat. Podeu veure un exemple d'aquestes imatges a baixa qualitat a la Figura 5.1.

L'algorisme de *Trianta*, però, si l'encadenem després d'algun dels altres algorismes que suavitzen la imatge, permet recuperar bé moltes cantonades accentuant-les. Cal mencionar que aquest algorisme no manté els coeficients dins l'espai de quantització, i per això després d'aplicar-lo, recomanem una *projecció sobre el QCS*. Podeu veure un exemple del seu ús en compressió a molt baixa qualitat a la Figura 5.2.

Respecte als plans de color, que en el cas de poca qualitat segurament hauran estat delmats i codificats amb unes taules de quantització de valors molt alts, donaran grans blocs diferenciats de color a la imatge. El que ens ha semblat més correcte ha estat utilitzar algorismes que suavitzin molt, com els de *Nosratinia* i *HP*. Utilitzant l'escalador ponderat per lluminositat ens disminuirà els vessaments de color importants que podem trobar en un cas com aquest.

Si l'ús excessiu d'algorismes suavitzants ens ha reduït el contrast de la imatge, podem arreglar-ho a la imatge final mitjançant un programa de retoc fotogràfic.



Figura 5.1: Codificació/descodificació de la imatge Lena de 256x256 píxels: *a)*original, *b)*JPEG qualitat 40 - PSNR 44,09 dB, *c)*descodificació amb *Froment* [2] (pesos de Froment, 8 iteracions, mida inicial de pas 1) i posterior esmolat amb GIMP (Filtre “afila”, amb afilament 30) - PSNR 44,99 dB, i *d)*versió JPEG2000 (amb *jasper* versió 1.701.0) amb la mateixa relació de compressió que la JPEG - PSNR 68,89 dB.



Figura 5.2: Codificació/descodificació de la imatge Lena de 256x256 píxels a molt baixa qualitat: *a)*original, *b)*JPEG qualitat 10 - PSNR 26,04 dB, *c)*descodificació amb *Froment* [2] (pesos de Froment, 12 iteracions, mida inicial de pas 1) - PSNR 28,92 dB, i *d)*descodificació amb *Froment* (com abans) seguit de *Trianta* [28] ($\sigma_{lf} = 2, \sigma_{hf} = 2,5, \sigma_{WSMM} = 1, \mu = 5, T = 500$) i *projecció sobre QCS* - PSNR 27,90 dB

5.2 Efecte d'ones superior al de blocs

Quan els plans es codifiquen a la qualitat sovint oferta per defecte als programes de manipulació gràfica (75 a la `libjpeg`), els efectes d'ones sovint superen els de blocs. En aquests casos, *Nosratinia* i *HP* no els eliminen del tot, i els algorismes iteratius de *Froment*, *O'Rourke* i *Robertson* ofereixen bons resultats, tot i que amb masses iteracions o passos per iteració massa grans (més de 10) suavitzarem la imatge més del compte. Per exemple, quan apliquem excessivament *O'Rourke* o *Robertson* (molt similars), els gradients d'intensitat de la imatge sofriran una mena de reducció de color. Les petites diferències de tonalitat desapareixeran, deixant només les cantonades més marcades.

També hem comprovat l'efectivitat de restaurar el pla de lluminositat amb el *filtre Gaussià selectiu* del GIMP, i involucrar el resultat al mig de la cadena de descodificació. Si es tracta d'una codificació d'alta qualitat (amb intervals de quantització prou petits), aquest algorisme no ens suavitzarà zones d'altres freqüències. Això sí, caldrà aplicar una projecció sobre el QCS abans de l'escalador per a obtenir una versió més fidel a la codificació original.

Respecte als plans de color, l'escalador de color que pondera segons la lluminositat ens dona els millors resultats dels tres que hi ha implementats en totes les imatges que hem provat, també en el cas de JPEG d'alta qualitat si és que hi ha delmat de color. Podem veure un exemple a la Figura 5.3

En el cas de que algunes textures es perdin massa pel suavitzat d'alguns algorismes, podem utilitzar l'algorisme *HContrast* (Apèndix B.1) per a intentar recuperar-les. Podem veure un exemple del seu ús a la Figura 5.4. A la mateixa figura també hi podem distingir clarament les millores de color gràcies a l'algorisme *O'Rourke* i el *LumScaler*.

5.3 Dibuixos

Tot i que la codificació JPEG va ser pensada per a fotografies, hi ha gent que l'ha utilitzat per a conservar i distribuir imatges artificials com dibuixos o logotips. En aquests casos el soroll ocasionat per la pèrdua d'informació pot ser més molest que en una fotografia codificada amb els mateixos paràmetres. Fins el 1996 els dibuixos artificials a través d'Internet estaven dominats pel format GIF, ja que fins el 1995 aquest format va ser de lliure ús. Des de llavors, l'empresa Unisys va decidir fer respectar la patent inherent en el GIF. A més, el format GIF limitava la imatge a 256 colors d'una paleta de 24 bits, amb un dels colors representant la transparència (útil en la composició d'imatges en una web). Quan Internet es va popularitzar, poc programari

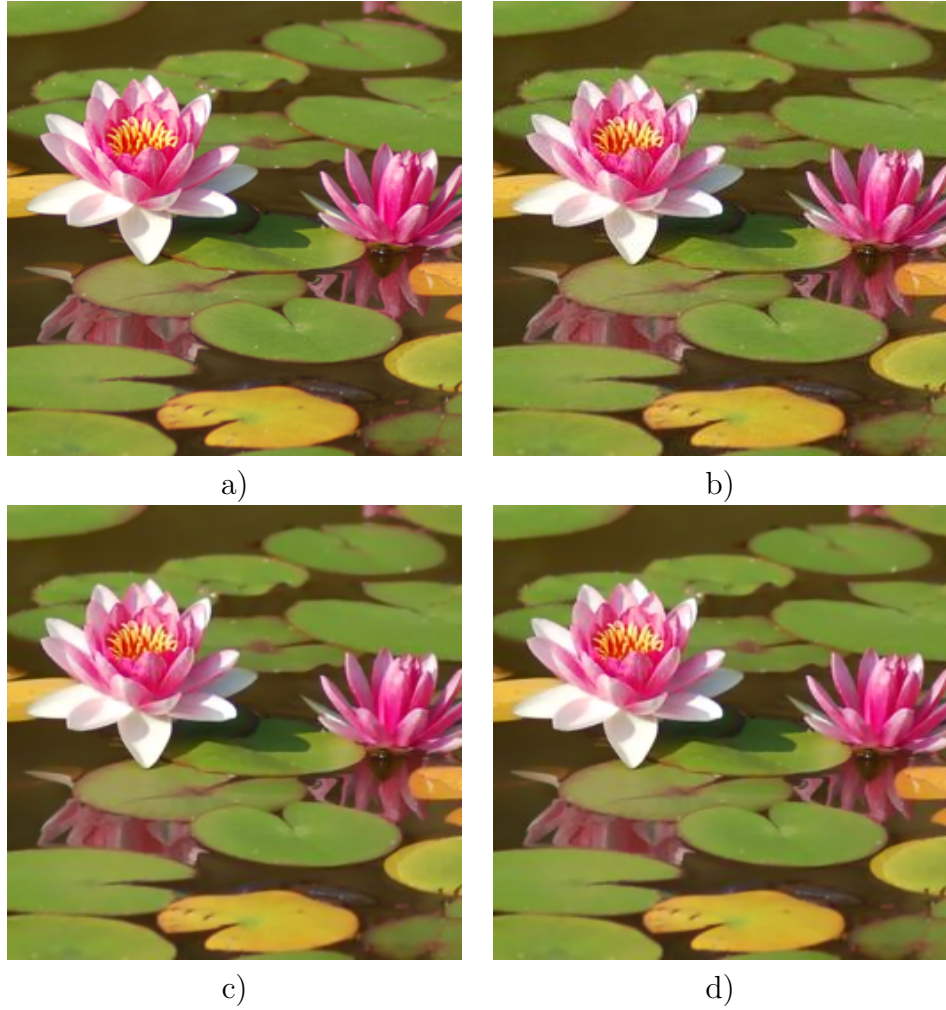


Figura 5.3: Codificació/descodificació d'una fotografia d'un nenúfar de 256x256 píxels, amb la qualitat per defecte de la `libjpeg`. *a)*original, *b)*JPEG qualitat 75, delmat 4:2:0 - PSNR R/G/B 38,47/47,33/33,87 dB, *c)*descodificació amb *Froment* [2] (pesos de Froment, 6 iteracions, mida inicial de pas 1) i escalador *Fancy* - PSNR R/G/B 41,26/46,43/37,52 dB, i *d)*descodificació amb *Froment* (com abans) seguit de l'escalador *LumScale* - PSNR R/G/B 42,12/47,92/39,82 dB



Figura 5.4: Descodificació d'una fotografia publicada a Internet, de 256x256 píxels. *a)*publicació original, *b)*descodificació amb Froment [2] (pesos de Froment, 5 iteracions, mida inicial de pas 1), *c)*descodificació amb Froment com abans, més O'Rourke [20] (10 iteracions, $T = 0,1$, mida inicial de pas 1) als canals de color, i l'escalador LumScaler ($\sigma_h = \sigma_v = 0,25$, Apèndix B.2). *d)*descodificació amb Froment i O'Rourke com abans, amb HContrast (Llindar 0,05 i finestra de 7x7), i LumScaler també com abans.

suportava el format PNG, sobretot els navegadors web¹. Llavors, bé pel límit de colors, bé per costum, molta gent ha conservat imatges artificials en JPEG.

Els algorismes de recuperació analitzats donen resultats especialment bons en aquest tipus d'imatges. Ens referim a imatges amb un contrast molt alt, amb pocs colors i les cantonades molt ben marcades. Precisament això fa que ens trobem més vessaments de color, i un acusat efecte d'ones. Els mètodes iteratius, especialment *O'Rourke* i *Robertson*, combinats amb l'escalador ponderat per lluminositat, ofereixen recuperacions excel·lents en aquestes condicions. En podem veure un exemple a les Figures 5.5 i 5.6.



Figura 5.5: a) Logotip d'un campionat del món codificat amb JPEG, agafat d'Internet (<http://www.futsalcat.com/seleccioabsoluta/copamon/LOGO%20COPA%20DEL%20MON%202007.jpg>). b) Recuperació segons una baula d'*O'Rourke* (7 passos, mida de pas de 1, $T = 0,1$) per canal, i *LumScaler*.

¹Per exemple, el Microsoft Internet Explorer suporta PNGs només des de la versió 5, i el canal alfa de transparència del PNG des de la versió 7.

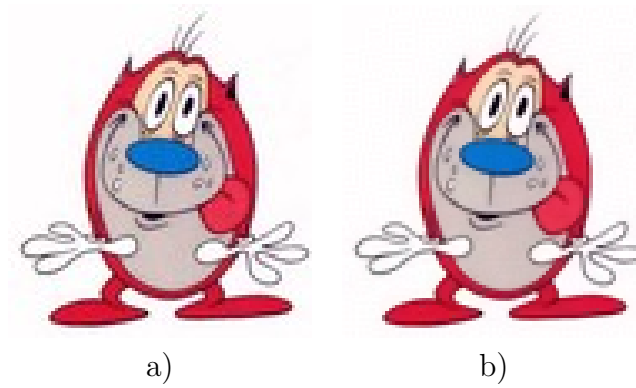


Figura 5.6: *a)* Personatge dibuixat (Stimpy, de John Kricfalusi) codificat amb JPEG, agafat d'Internet (<http://www.tvacres.com/images/stimpy.jpg>). *b)* Recuperació segons una baula d'O'Rourke (5 passos, mida de pas de 1, $T = 0,1$) per canal, i *LumScaler*.

Capítol 6

Conclusions i treball futur

La popularitat de la codificació JPEG va ser el principal motiu per a preocupar-nos per la seva descodificació. Moltes fotografies ens són accessibles únicament en aquest format. L'evidència de que el procés de descodificació es podia millorar, i que en prou feines existissin eines informàtiques a aquest efecte, ens semblen motius suficients per a dedicar aquest projecte al desenvolupament d'un descodificador JPEG interactiu com el que hem presentat. En el procés d'investigació de les tècniques disponibles, hem recopilat prou informació com per oferir un resum prou complet del camp de la restauració d'imatges JPEG, i hem implementat alguns dels algorismes més interessants dins aquest descodificador interactiu.

En aquest document presentem tant els problemes de la codificació JPEG com l'estat de l'art de la seva descodificació, convertint-se en el document ideal pels catalano-parlants que vulguin introduir-se en aquest camp. El programa informàtic representa una plataforma còmoda per acostar les investigacions en aquest camp a públic més profà. Nosaltres hem decidit aturar el desenvolupament de l'aplicació de manera que ens permetés elaborar aquesta investigació, i alhora usuaris ja se'n poguessin beneficiar. A partir d'aquest punt hem considerat aquest projecte acabat, i hem publicat l'aplicació a Internet. Això no vol dir que el programa no es pugui millorar, ni que deixem de treballar en ell; ara altres programadors poden afegir-hi els seus algorismes, i a més podem considerar la resposta dels usuaris.

Hem publicat l'aplicació amb C++ sota la llicència GPL versió 2¹; en part perquè volem que la seva evolució continuï lliure i pública, i en part perquè utilitzem la llibreria Qt² per a la interfície gràfica. Aquesta tecnologia ens permet publicar una base de codi única per a Linux/X-Windows, Macintosh OS X i Microsoft Windows. A més el programa està preparat per a ser

¹<http://www.gnu.org/licenses/old-licenses/gpl-2.0.html>

²<http://trolltech.com/>, que ens exigeix seguir la llicència GPLv2.

traduït a diferents idiomes, havent publicat de moment la interfície en català, esperanto i anglès.

Abans de la publicació del programa, l'hem mostrat a un petit nombre d'individus que ens han corroborat consideracions del seu disseny: els interessava més comparar les diferents restauracions a cop d'ull, independent de qualsevol mesura matemàtica. I diferents usuaris (amb les seves circumstàncies) solen arribar a diferents restauracions, cadascú preferint la seva. Entre els comentaris dels usuaris i alguns investigadors hi preval el de sorpresa i alegria de que per fi existeixi un programa com aquest. Per tant, considerem que hem assolit els objectius de fer arribar tecnologia de restauració JPEG recent dels laboratoris als usuaris, i hem recopilat l'estat de l'art d'aquest camp amb la plusvàlua de resultats pràctics gràcies a les pròpies implementacions d'algorismes.

Apèndix A

Transformada DCT

Al codificar els blocs de 8x8 píxels de cada pla de la imatge, els apliquem una transformada que es coneix amb el nom de FDCT (*Forward Discrete Cosine Transform*). Donats els 64 valors d'intensitat de cada píxel, obtenim 64 nous valors del domini DCT. El primer d'aquests nous 64 valors l'anomenem el coeficient DC (de freqüència zero, component contínua), i els 63 restants coeficients AC (de freqüència diferent de zero). Mitjançant la transformada IDCT (*Inverse DCT*) podem tornar a recuperar els valors dels píxels, donats els 64 valors del domini DCT. Per a un bloc de píxels s_{xy} ($x, y = 0, 1, \dots, 7$), la FDCT està establerta com ([13]):

$$S_{uv} = \frac{1}{4} C_u C_v \sum_{x=0}^7 \sum_{y=0}^7 s_{xy} \cos \frac{(2x+1)u\pi}{16} \cos \frac{(2y+1)v\pi}{16}$$

donant els 64 valors del domini freqüencial S_{uv} ($u, v = 0, 1, \dots, 7$) i la IDCT com:

$$s_{xy} = \frac{1}{4} C_u C_v S_{uv} \sum_{x=0}^7 \sum_{y=0}^7 \cos \frac{(2x+1)u\pi}{16} \cos \frac{(2y+1)v\pi}{16}$$

on

$$C_m = \begin{cases} \frac{1}{\sqrt{2}} & \text{si } m = 0 \\ 1 & \text{altrament} \end{cases}$$

Podem veure gràficament les bases de la transformada per coeficient, i la influència gradual de les freqüències més altes a la Figura 1.4, pàg. 14.

Apèndix B

Algorismes en detall

B.1 Manteniment de textures

La majoria dels mètodes de millora de pla estudiats utilitzen alguna mena de suavitzat per eliminar, entre altres, els efectes de bloc. Si aquest suavitzat s'aplica a zones on hi ha textures d'alta freqüència, el resultat pot empitjorar respecte a una descodificació convencional. En la majoria de casos això es traduirà en un augment de la PSNR, però ja hem explicat a l'apartat 2 com la PSNR no ens serveix com a mesura de qualitat en zones d'altres freqüències.

És per això que hem proposat un mètode adaptatiu per a no empitjorar aquest tipus de zones. Ens basem en que si suavitzem una zona d'altres freqüències, el seu contrast es veurà reduït. Per cada punt de la imatge, considerem que si hem reduït molt el contrast local respecte la descodificació convencional, val més utilitzar el valor del píxel d'aquesta última. Avaluem tota la imatge segons el contrast local per cada punt i,j de la imatge. El contrast el valorem segons Michelson:

$$C(i,j) = \frac{I_{\max}(i,j) - I_{\min}(i,j)}{I_{\max}(i,j) + I_{\min}(i,j)}$$

A l'entorn local:

$$I_{\max}(i,j) = \max_{k,l \in S} y(k,l) \quad (\text{B.1})$$

$$I_{\min}(i,j) = \min_{k,l \in S} y(k,l) \quad (\text{B.2})$$

$$(\text{B.3})$$

on $y(k,l)$ és el valor associat al punt k,l de la imatge, i S el conjunt de punts d'una finestra quadrada de 3x3, 5x5, 7x7, ... a l'entorn de i,j .

Calculem doncs el contrast local al pla restaurat $C(i,j)$, i al pla descodificat de manera convencional $C'(i,j)$ per cada i,j de la imatge. Llavors, si $y_1(i,j)$ fa referència a un punt del pla restaurat, i $y_c(i,j)$ a un del pla convencional, el nou punt restaurat $y_2(i,j)$ ve determinat per:

$$y_2(i,j) = \begin{cases} y_1(i,j) & \text{si } C'(i,j) - C(i,j) > \epsilon \\ y_c(i,j) & \text{altrament} \end{cases}$$

L'usuari pot escollir tant el llindar ϵ com la mida de la finestra de S .

B.2 Escalat dels plans de color ponderats amb la lluminositat

A l'article [26] proposen el ponderat per a imatges que tenen els plans de color delmats co-situats (*cosited*). Això vol dir que els punts del pla de color delmat coincideixen amb la posició de punts del pla sense delmar. A les imatges JPEG, tal com diu l'especificació del JFIF [9], els plans delmats tenen els punts inter-situats (*intersited*) respecte al pla més gran (vegeu la Figura B.1. Això fa una mica més complexes els algorismes.

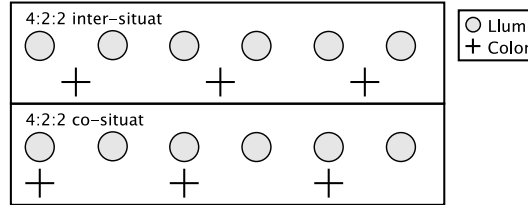


Figura B.1: Diferència entre un pla de color delmat co-situat i un d'inter-situat pel cas 4:2:2.

B.2.1 Delmat 4:2:2

En el cas 4:2:2 el valor de cromà del pla delmat, a la codificació, ha estat calculat segons la mitjana aritmètica dels dos punts originals més propers. Aquest algorisme proposa fer el mateix amb el pla de lluminositat, com si el delméssim. Del pla delmat anomenem Y_1 al punt més proper al nostre punt d'interès, i Y_2 al segon més proper. Com que el pla de lluminositat el tenim també sense delmar, sabem el valor del punt d'interès Y_o . Llavors, podem utilitzar la relació $\frac{Y_o - Y_1}{Y_2 - Y_1}$ per a ponderar el pla de color. Podem veure-ho gràficament a la Figura B.2

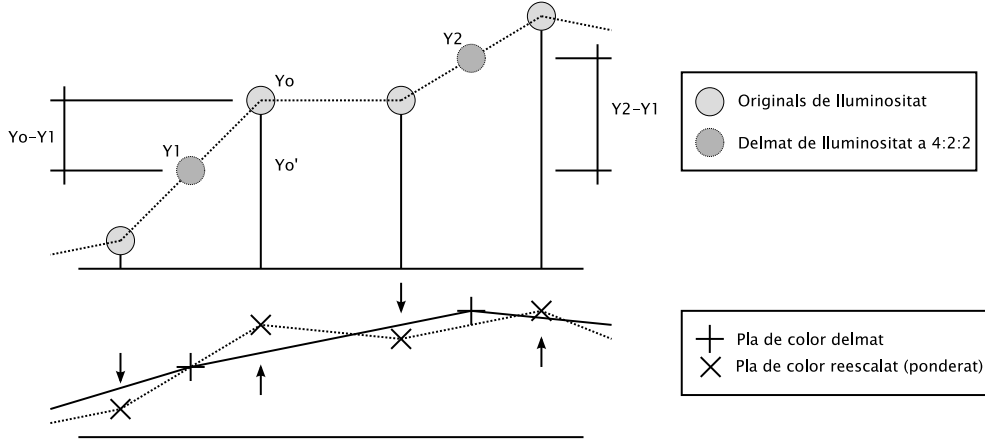


Figura B.2: Escalat de color de 4:2:2 a 4:4:4 amb ponderació per lluminositat. Calculem la versió delmada de la lluminositat, i obtenim Y_1 i Y_2 , on Y_1 és el punt delmat més proper a Y_0 , i Y_2 el segon més proper. Veiem que la interpolació lineal entre Y_1 i Y_2 ens donaria un punt inferior a Y_0 - i aquesta és la correcció que es fa palesa al pla de color. Les creus dretes ens mostren la versió delmada, i ponderant el pla de color amb $\frac{Y_0 - Y_1}{Y_2 - Y_1}$ aconseguim també l'anàleg a Y_0 més alt que si utilitzéssim interpolació lineal.

Ara podem reescriure l'algorisme en funció de cada punt de la imatge (delmat i sense delmar), i limitar la ponderació segons un paràmetre ϵ_h . En general, per cada fila tenim $1..N$ columnes de pla de color i $1..2N$ columnes de pla de lluminositat. Llavors, per cada punt i del pla de color obtenim:

$$Y(2i) = C(i) + (C(i+1) - C(i)) w(Y(2i), Y_m(2i-1), Y_m(2i+1)) \quad (B.4)$$

$$Y(2i+1) = C(i+1) + (C(i) - C(i+1)) w(Y(2i+1), Y_m(2i+1), Y_m(2i-1)) \quad (B.5)$$

on el pes $w(p, n, f)$ i el delmat de lluminositat $Y_m(j)$ els definim així:

$$w(p, n, f) = \begin{cases} 0,25 + \epsilon_h & \text{si } w'(p, n, f) > 0,25 + \epsilon_h \\ 0,25 - \epsilon_h & \text{si } w'(p, n, f) < 0,25 - \epsilon_h \\ w'(p, n, f) & \text{altrament} \end{cases} \quad (B.6)$$

$$w'(p, n, f) = \frac{p - n}{f - n} \quad (B.7)$$

$$Y_m(j) = \frac{Y(j) + Y(j+1)}{2} \quad (B.8)$$

on l'usuari pot escollir ϵ_h .

B.2.2 Delmat 4:2:0

L'algorisme de 4:2:2 involucra dos punts Y_1 i Y_2 per cada punt de la imatge final. En canvi, en el cas 4:2:0 tenim quatre punts involucrats, i ens cal adaptar l'algorisme a aquests quatre punts. Com bé sabem, tenim tant delmat horitzontal com vertical. Així, la nostra adaptació consisteix en primer considerar el cas horitzontal i després el vertical, per cada punt de la imatge.

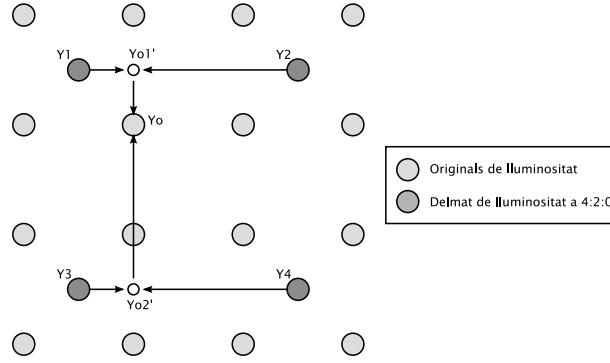


Figura B.3: Escalat de color 4:2:0 a 4:4:4 ponderant segons la lluminositat.

Seguint l'esquema de la Figura B.3, i amb l'objectiu de determinar el color al punt Y_0 , considerem els punts $Y_{1..4}$ com els punts ordenats segons el més proper, el segon més proper en horitzontal, el tercer més proper en vertical, i el quart més proper. Utilitzant la tècnica del cas de 4:2:2, considerem el punt temporal Y_{o1} i en calculem la croma ponderada segons Y_1 i Y_2 (amb els càlculs pertinents de la lluminositat) i ϵ_h . Fem el mateix amb el punt Y_{o2} , Y_3 , Y_4 , i la mateixa ϵ_h . Finalment, a partir d'aquestes noves cromes dels punts Y_{o1} i Y_{o2} , i d'unes versions de la lluminositat de Y_{o1} i Y_{o2} (segons la mitjana dels seus punts de lluminositat més propers), fem la ponderació per aconseguir Y_0 , aquest cop segons un paràmetre ϵ_v també configurable per l'usuari.

Bibliografia

- [1] Amjed S. Al-Fahoum i Ali M. Reza. Combined edge crispiness and statistical differencing for deblocking jpeg compressed images. *IEEE Transactions on Image Processing*, 10(9):1288–1298, September 2001.
- [2] François Alter, Sylvain Durand i Jacques Froment. Adapted total variation for artifact free decompression of jpeg images. *Journal of Mathematical Imaging and Vision*, 23:199–211, 2005.
- [3] Tao Chen, Hong Ren i Bin Qiu. Adaptive postfiltering of transform coefficients for the reduction of blocking artifacts. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(5):594–602, May 2001.
- [4] François-Xavier Coudoux, Marc Gazelet i Patrick Corlay. An adaptive postprocessing technique for the reduction of color bleeding in dct-coded images. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):891–897, January 2004.
- [5] Ricardo L. de Queiroz. Processing jpeg-compressed images and documents. *IEEE Transactions on Image Processing*, 7(12):1661–1672, December 1998.
- [6] Wenfeng Gao, Coskun Mermer i Yongmin Kim. A de-blocking algorithm and a blockiness metric for highly compressed images. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(12):1150–1159, December 2002.
- [7] Richard L. Gregory. *Eye and Brain – The Psychology of Seeing*. Princeton University Press, cinquena edició, 1997.
- [8] The Independent JPEG's Group. The libjpeg jpeg library. Software, March 1998.
- [9] Eric Hamilton. *JPEG File Interchange Format, Version 1.02*, September 1992. Version 1.02.

- [10] ITU. *Information Technology – Digital Compression and Coding of Continuous-Tone Still Images – Requirements and Guidelines*, September 1992. ITU. ISO/IEC 10918-1:1993(E). CCITT Rec. T.81 (1992 E).
- [11] J. Jung, M. Antonini i M. Barlaud. Optimal jpeg decoding. *International Conference on Image Processing 1998. ICIP 98. Proceedings.*, 1:410–414, October 1998.
- [12] Sung Deuk Kim, Jaeyoun Yi, Hyun Mun Kim i Jong Beom Ra. A deblocking filter with two separate modes in block-based video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(1):156–160, February 1999.
- [13] Weidong Kou. *Digital Image Compression – Algorithms and Standards*. Kluwer Academic Publishers, primera edició, 1995.
- [14] Kee-Koo Kwon, Byung-Ju Kim, Suk-Hwan Lee, Jong-Won Lee, Seong-Geun Kwon i Kuhn-Il Lee. Postprocessing algorithm in block-based codec images using wavelet transform and adaptive mlp. *International Conference on Multimedia and Expo 2002. ICME '02. Proceedings.*, 1:165–168, 2002.
- [15] Y. L. Lee, H. C. Kim i H. W. Park. Blocking effect reduction of jpeg images by signal adaptive filtering. *IEEE Transactions on Image Processing*, 7(2):229–234, February 1998.
- [16] Alan W.-C. Liew i Hong Yan. Blocking artifacts suppression in block-coded images using overcomplete wavelet representation. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(4):450–461, April 2004.
- [17] Shizhong Liu i Alan C. Bovik. Efficient dct-domain blind measurement and reduction of blocking artifacts. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(12):1139–1149, December 2002.
- [18] Shigenobu Minami i Avidesh Zakhori. An optimization approach for removing blocking effects in transform coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 5(2), April 1995.
- [19] Aria Nosratinia. Enhancement of jpeg-compressed images by re-application of jpeg. *Internet Publication*, October 2002.
- [20] Thomas P. O’Rourke i Robert L. Stevenson. Improved image decompression for reduced transform coding artifacts. *IEEE Transactions on Circuits and Systems for Video Technology*, 5:490–499, December 1995.

- [21] Tuan Q. Pham i Lucas J. van Vliet. Blocking artifacts removal by a hybrid filter method. *11th Annual Conf. of the Advanced School for Computing and Imaging*, pàgines 372–377, June 2005.
- [22] K. R. Rao i P. C. Yip, editors. *The transform and data compression handbook*, capítol 1. CRC Press LLC, primera edició, 2001.
- [23] Mark A. Robertson i Robert L. Stevenson. Dct quantization noise in compressed images. *Draft*, February 2004.
- [24] Ramin Samadani, Arvind Sundararajan i Amir Said. Deringing and deblocking dct compression artifacts with efficient shifted transforms. *IEEE International Conference on Image Processing (ICIP)*, Singapore, October 2004.
- [25] Hamid Rahim Sheikh i Alan C. Bovik. Image information and visual quality. *IEEE Transactions on Image Processing*, 15(2):430–444, February 2006. <http://live.ece.utexas.edu/research/quality/VIF.htm>.
- [26] Hiroshi Sugita i Akira Taguchi. Chrominance signal interpolation of yuv 4:2:0 format color images. *Electronics and Communications in Japan, Part 3*, 89(9), 2006.
- [27] G. A. Triantafyllidis, M. Varnuska, D. Sampson, D. Tzovaras i M. G. Strintzis. An efficient algorithm for the enhancement of jpeg-coded images. *Computers and Graphics*, 27:529–534, 2003.
- [28] George A. Triantafyllidis, Dimitrios Tzovaras i Michael Gerassimos Strintzis. Blocking artifact detection and reduction in compressed data. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(10):877–887, October 2002.
- [29] Zhou Wang i Alan C. Bovik. Mean squared error: Love it or dump it? – a new look at signal fidelity measures. *Draft*, December 2006.
- [30] Zhou Wang, Hamid R. Sheikh i Alan C. Bovik. No-reference perceptual quality assessment of jpeg compressed images. *IEEE International Conference on Image Processing*, September 2002.
- [31] Chaminda Weerasinghe, Alan Wee-Chung Liew i Hong Yan. Artifact reduction in compressed images based on region homogeneity constraints using the projection onto convex sets algorithm. *IEEE Transactions on Image Processing*, 12(10):891–897, October 2002.

- [32] Wikipedia. Jpeg — Wikipedia, the free encyclopedia, 2007. [En línia, accés el 25 d'octubre de 2007].
- [33] H. R. Wu i M. Yuen. A generalized block-edge impairment metric for video coding. *IEEE Signal Processing Letters*, 4(11), November 1997.
- [34] Yoingyi Yang i Nikolas P. Galatsanos. Removal of compression artifacts using projections onto convex sets and line process modeling. *IEEE Transactions on Image Processing*, 6(10):1345–1357, October 1998.
- [35] Yoingyi Yang, Nikolas P. Galatsanos i Aggelos K. Katsaggelos. Regularized reconstruction to reduce blocking artifacts of block discrete cosine transform compressed images. *IEEE Transactions on Circuits and Systems for Video Technology*, 3(6):421–432, December 1993.