Implementació d’un esquema de proteccció del copyright per a imatges

Autor: Joan Urrutia Farré
Consultor : Francesc Sebé Feixas

Universitat Oberta de Catalunya

Data: 21/06/2002

INDEX

1.- Introducció 6
2.- Imatge Digital 8
2.1.- Què és la imatge digital 8
2.1.1.- Vector gràfic i mapa de bits 8
2.1.2.- Models de color 8
2.1.3.- Color veritable i Paleta de colors 9
2.1.4.- Tipus d'imatge digital 9
2.2.- Compressió d'imatges 11
2.3.- El format d'imatge JPEG 12
2.3.1.- Compressió JPEG enfront altres formats comprimits 12
2.3.2.- Avantatges i desavantatges de la compressió amb o sense pèrdua d'informació 12
2.3.3.- Tipus de compressió JPEG 13
2.3.3.1.- Seqüencial 13
2.3.3.2.- Progressiu 13
2.3.3.3.- Sense pèrdua 14
2.3.4.- Anàlisi detallat del JPEG usat en el TFC 14
2.3.4.1.- Nocions bàsiques 14
2.3.4.1.1.- Ordenació de bits 14
2.3.4.1.2.- Freqüència de mostreig 14
2.3.4.1.3.- Descripció del procés 15
2.3.4.1.4.- Escombrat a un sol component o a més d'un 15
2.3.4.2.- Codi de Huffman 16
2.3.4.2.1.- Idea general 16
2.3.4.2.2.- Codi de Huffman en JPEG 17
2.3.5.- Creació d'un arxiu JPEG seqüencial 18
2.3.5.1.- Paràmetres de compressió 18
2.3.5.2.- Descompressió 18
3.- Fonaments teòrics del treball 19
3.1.- Fonaments de protecció del copyright 19
3.2.- Processos de marcatge i recuperació de la marca 20
3.3.- Propietats dels esquemes de protecció del copyright 21
3.3.1.- Propietats generals 21
3.4.- Classificació dels esquemes de protecció del copyright 22
3.4.1.- Watermarking 22
3.4.2.- Fingerprinting 23
3.4.2.1.- Fingerprinting simètric 23
3.4.2.2.- Fingerprinting asimètric 24
3.4.2.3.- Fingerprinting anònim 25

4.- Esquema de watermarking implementat 26
4.1.- Algorisme d'inserció de marca 26
4.2.- Algorisme de recuperació de marca 28
4.3.- Robustesa de l'esquema contra atacs de compressió JPEG 29
4.3.1.- Experiments que s'han realitzat 29
4.3.2.- Resultats de les proves 30
4.4.- Detalls d'implementació 34
4.4.1.- Llenguatge de programació 34
4.4.2.- Sintaxi dels programes 34
4.4.3.- Detalls de programació 35
5.- Conclusions 37
6.- Bibliografia 38
7.-Annexos
Annex A . Algorisme de procés de marcatge 39
Annex B . Algorisme de recuperació de marca 41
Annex C . Imatges fetes servir al treball 43

PRELIMINARS

Es pretén implementar un sistema de detecció de còpia per protegir el copyright d'imatges digitals en format JPEG.

Aquest sistema està composat per dos algorismes:

- Inserció de marca: aquest algorisme pren una imatge en format JPEG com a entrada i hi insereix una marca de forma imperceptible.

- Recuperació de marca: aquest algorisme pren una imatge en format JPEG i en recupera la marca que se li ha inserit mitjançant l'algorisme anterior.

Un cop implementats els dos algorismes, es farà un estudi sobre la robustesa del sistema enfront a manipulacions de compressió.

El treball consta de dues parts principals:

- Estudi del concepte de protecció del copyright i el funcionament general d'un sistema de detecció de còpia
- Implementació i avaluació d'un sistema d'aquest tipus aplicat a imatges en format JPEG.

OBJECTIUS

Es vol implementar un programa que permeti la inclusió i posterior recuperació de marques en imatges en format JPEG. Posteriorment s'estudiarà la robustesa del sistema implementat en front a manipulacions de compressió.

Els objectius d'aquest treball són:

- Entendre la importància de la protecció del copyright en el comerç electrònic
- Estudiar el funcionament i conceptes fonamentals dels sistemes de detecció de còpia
- Estudiar el format d'imatge JPEG
- Implementar els algorismes d'inserció i recuperació de marca proposats per imatges en format JPEG
- Estudiar la robustesa del sistema implementat en front a manipulacions de compressió.

1-Introducció

El passat dijous 7 de març mentre llegia el suplement (Cyberp@is) de El País, una frase ressaltada en negreta va captar la meva atenció: "***Las compañías de ocio quieren vetar cualquier tecnología que pueda usarse para copias ilegales, aunque la tecnología sea totalmente legal***". Aquesta frase formava part d'un article d'en Francis Pisani titolat *Nueva querella entre antiguos y modernos*.

El primer que et passa pel cap és que l'autor ha exagerat una mica el problema, però una vegada llegit l'article t'adones de la importància de la problemàtica que esmenta. En aquests moments un terrible dilema està davant la taula d'una comissió del Senat dels EEUU, i la seva decisió té tota la pinta de ser salomònica, ja que han d'elegir entre arruïnar la indústria de l'espectacle o paralitzar la capacitat d'innovació tecnològica.

Fa uns quants anys va tenir lloc una querella dels fabricants de discos contra Napster, que després de molts litigis van guanyar, sent portada en nombrosos diaris. D'aquest tema es va escriure molt, però no va anar més lluny. Ara, si mirem el resultat d'allò ens adonem que han sortit altres empreses: Morpheus, Kazaa, Grokster.. que fan el mateix . De seguida t'adones de que si les companyies discogràfiques que van guanyar una batalla i van acabar perdent la guerra, volen començar una batalla contra la indústria informàtica, un gegant en aquests moments, el resultat d'aquesta es pot predir fàcilment.

El fort creixement que està experimentant el comerç electrònic està obrint noves perspectives per a tot tipus d'empreses, entre elles les discogràfiques, que els hi reportarà una disminució de costos, i, per tant, majors beneficis. Si una companyia discogràfica pot vendre directament el seu producte al client sense necessitat de vendre'l amb el suport físic (CD, DVD) i estalviar-se els d'intermediaris, aleshores el client obtindrà el mateix producte a molt millor de preu. Així tenim que la companyia de discos pot vendre moltes més unitats i, fins i tot guanyar molt més diners. Al client, el producte li resultarà a un preu molt més assequible. Però per a que això pugui ser factible ha de ser possible que el client pugui gravar el producte comprat en un CD o DVD mitjançant la tecnologia i els aparells que ara les companyies de discos volen prohibir. No seria millor que en lloc d'una lluita entre les companyies tecnològiques i informàtiques vers les discogràfiques, hi hagués una certa col·laboració que beneficiaria principalment a les segones que són les que tenen més a guanyar?

Aquest sistema que tenen les companyies discogràfiques de vendre els seus productes mitjançant Internet, fent que tingui una reducció de costos de producció, distribució i venda, i donar-li al client un producte molt barat i amb una bona qualitat, té el risc de la pirateria. I qui millor que les noves tecnologies i empreses del sector informàtic per ajudar a les companyies de discos enfront un enemic comú?

És per això que la proposta del senador Hollings que s'esmenta en l'article en què "*es demana col·locar un senyal en els productes, i en la computadora un mecanisme que impedeixi la lectura de qualsevol CD o DVD o vídeo que no tingui el senyal*",

considero que no es prou encertada per tal de fomentar una bona entesa entre les diferents companyies.

Crec que hi ha moltes maneres de trobar solucions amb les que no calgui fer modificacions en el hardware dels computadors. Es podria aprofitar els avenços a nivell de software que apareixen com a conseqüència dels reptes plantejats pel correu electrònic, com poden ser la implementació de sistemes de pagament electrònic, desenvolupament de mètodes de protecció de copyright electrònic, estudi de sistemes d'agents intel·ligents per a la gestió de la informació, etc.

El que tenen molt clar tant les companyies discogràfiques com les de Silicon Valley és que cal prevenir la distribució il·legal amb la inclusió de nous elements de seguretat en la venda de productes en suport electrònic. No així la manera com fer-ho. Els intents encaminats a la protecció de productes per tal d'evitar la seva còpia, és molt difícil. Així doncs, si es permet que les companyies discogràfiques controlin tota la maquinària que permet copiar arxius, mitjançant la col·locació en cada aparell dels mecanismes que creguin més oportú, aleshores haurien aconseguit el control en un mercat (el de l'intercanvi d'arxius) que és una font considerable d'ingressos. La qual cosa les empreses del sector no tenen la més remota intenció de permetre. Sobretot ara que hi han altres maneres d'afrontar aquest problema i que podem extreure de les paraules d'en Leslie Vadasz, vice-president d'Intel: "*L'enfocament del debat no hauria de ser la protecció del contingut sinó la del consumidor*".

Així doncs el que cal fer és encaminar la recerca a realitzar en aquest àmbit cap al concepte de 'detectar' en lloc del que fins ara es dirigien tots els esforços que era el 'evitar'. Una vegada que ha quedat demostrat que realment és impossible evitar que es facin còpies il·legals, com s'ha pogut veure en l'intent d'acabar amb la distribució de pel·lícules i imatges per part de Napster, o en l'intent d'ara d'intentar controlar el consum mitjançant el control de la indústria informàtica que ara s'està discutint en una comissió del senat, el que cal fer és d'assegurar-se de què qui fa una còpia il·legal d'un producte podrà ser identificat.

L'article en el seu darrer paràgraf ens diu que "*aquesta batalla entre les companyies discogràfiques i les de suports informàtics està a punt de convertir-se en una nova querella entre antics i moderns*". Potser no, si podem contestar a la pregunta que es fa l'autor com a final del seu article:" *Com assegurar a la vegada la remuneració dels artistes i creadors de contingut i permetre que els consumidors gaudeixin totalment de les diferents tecnologies ?*". La idea general rau en incloure certes marques dins del producte que identifiquen al comprador, de manera que la còpia del producte impliqui també la còpia de les marques i per tant la còpia de la identitat de l'infractor.

En aquest TFC es fa una introducció al concepte de protecció del copyright mitjançant detecció de còpia i s'implementa un sistema basat en la inserció de marques per protegir el copyright en imatges digitals en format JPEG.

2 Imatge Digital

2.1 Què és la imatge digital?

A la gran majoria d'ordinadors, la imatge que ens apareix a la pantalla del monitor està formada per un gran nombre de punts quadrats que estan enganxats un al costat de l'altre i que ocupen la totalitat de la pantalla. Aquests punts són els anomenats píxels. Aquests píxels estan col·locats en ordre dins una cadena bidireccional i cadascun d'ells mostra un color determinat.

Els dibuixos i formes que veiem a la pantalla són el resultat d'ajustar el color de cadascun d'aquests píxels. Cada vegada tenim imatges de més resolució, ja que com més píxels té la pantalla aleshores els dents de serra que apareixen degut a la forma quadrada dels píxels es fa indistingible als ulls humans. Així hem passat en poc temps de monitors de poc més de 307,200 píxels (640x800) a més de 1.3 milions (1280x1024). El límit d'aquest nombre de píxels el determina la capacitat del monitor i l'adaptador de vídeo juntament amb la memòria de vídeo.

Un típic sistema de vídeo estaria format per un buffer on emmagatzemar la trama de píxels, una controladora de vídeo i un monitor. El buffer és un bloc de la memòria de vídeo que controla el color de cadascun dels píxels dels monitor. Així tenim que cada píxel té una localització determinada dins de la memòria, amb una mesura d'entre 1 i 32 bits. Així doncs, una aplicació pot modificar el color d'un píxel de la pantalla solament canviant els valors dels bits de la memòria que té assignat aquest.

2.1.1-Vector gràfic i mapa de bits

Per fer aparèixer les imatges a la pantalla del monitor hi ha dos mètodes d'operació per als formats dels arxius gràfics: el de vector gràfic i el mapa de bits. Un format de vector gràfic fa servir una sèrie de comandaments de dibuix per a representar la imatge. Aquest format no és útil per a representar fotografies o pintures, ja que necessitaria molts milers de comandaments de dibuix convertint-se en una tasca monumental, a més de necessitar molt temps per a mostrar la imatge en pantalla. Així ens trobem que la majoria d'imatges són representades mitjançant el format de mapa de bit. Aquest format representa les imatges com a cadenes bidimensionals on cada element de la cadena representa un color que serà col·locat en una posició específica dins la pantalla del monitor. Serà la representació per píxels. Aquests píxels en estar un al costat de l'altre i ser molt petits crea en l'ull humà una imatge uniforme.

2.1.2-Models de color

Per a representar el color en cada píxel el que es fa és fer servir un determinat model de color. Normalment es fa servir el mateix sistema que els televisors en color, que és el RGB. En el qual una combinació de diferents intensitats de llum roja, verda i blava conflueixen en un mateix punt per tal de donar-li un color determinat. Així doncs, amb un nombre de bits per cada color obtenim cadascun dels colors. Per

exemple, si cada píxel ocupa 24 bits, tenim 8 bits per color, el que ens dóna unes 256 tonalitats diferents per a cada color que fa possible combinant les diferents tonalitats dels colors primaris arribar fins a més de setze milions de colors diferents.

Un altre model de color és el YCbCr que es fa servir en algunes aplicacions per al processament d'imatges, entre elles les que es fan servir per emmagatzemar imatges de color en JPEG. Es basa en el model HSB, en què la lluminositat ve donada per la component Y. Aquesta component és la més important ja que representa la intensitat de la imatge. Al mateix temps correspon a la representació en escala de grisos de la mateixa. La component Cb ens dóna l'escala de blaus i Cr l'escala de vermells.

2.1.3 Color veritable i Paleta de color

Amb el sistema RGB, per la majoria d'ordinadors personals s'utilitza 8 bits per color. Aquestes imatges produïdes amb píxels de 24 bits és el que anomenem color veritable. Però també hi ha sistemes que utilitzen 12 bits en escales de gris per aplicacions mèdiques, o 12 bits per color amb JPEG o 16 amb PNG. Per tal d'assignar un color a un píxel hi ha dos mètodes que són els més utilitzats. El més simple consisteix en emmagatzemar el color veritable per cada píxel en la imatge comprimida. Aquest mètode té el problema que no tots els sistemes tenen una memòria de vídeo per a operar amb una imatge amb color veritable. Per això hi ha el mètode de la paleta de colors, en el qual se selecciona una mostra de colors entre tots els possibles. Així tenim una paleta de colors que seria una cadena unidimensional de 3 bytes per element que especificaria el color, i enlloc d'especificar directament aquest, el que fa el sistema és agafar el valor de cada píxel com un índex que apunta a la paleta de colors.

Les imatges amb 24 o més bits per píxel mai fan servir la paleta de color perquè necessitarien més bits per a especificar l'índex a la paleta que especificar el color directament. En canvi, els formats que fan servir la paleta de color han de fer servir valors de píxel amb 8 o menys bits per tal de reduir la mida de l'arxiu. La mida que més sovint s'usa per a les paletes de color és de 8 bits que ens dóna una varietat de 256 colors. Una imatge amb 4 bits per píxel requereix com a molt la meitat d'emmagatzematge que una imatge de 8 bits per píxel de la mateixa mida.

Si agafem els formats d'imatge podem constatar que tant el format BMP com el PNG poden fer servir tant paleta de color com veritable color. En canvi el format JPEG només fa servir color veritable i el format GIF sempre usarà paleta de color.

2.1.4 Tipus d'imatge digital

Hi ha molta varietat de formats d'imatge digital. Entre ells podem destacar els següents:

BMP

Aquest format especifica el color del píxel d'una forma diferent del mètode que es fa servir pels monitors per a representar el color, la qual cosa permet que sigui

utilitzat en qualsevol tipus de monitor. Una imatge en aquest format pot estar en color real quan fa servir 24 o més bits per píxel o bé fa servir una paleta de colors en les representacions en que usa 8 o menys bits per píxel.

Fa servir el mètode de compressió RLE, en què píxels consecutius amb el mateix valor són codificats amb 2 valors en hexadecimal: el primer indica el nombre de vegades que apareix consecutivament i el segon ens dóna el valor(color) del píxel.

GIF

Aquest format només fa servir paleta de colors i usa la tècnica de compressió anomenada LZ, que el que fa és mantenir un diccionari que conté les seqüències de píxels que van apareixent al llegir les dades. Així el format comprimit el que conté és una cadena de codis que representa les entrades d'aquest diccionari.

JPEG

És un dels millors per a emmagatzemar imatges. Només es fa servir imatges en color real. Pot fer servir diferents mètodes de compressió: RLE, Huffman i DCT. Segons el mètode que es faci servir, la compressió és amb pèrdua (la imatge que es treu al expandir-la no és exactament la mateixa que l'original) o sense pèrdua (s'obté la mateixa imatge). Té quatre tipus de compressió: seqüencial, progressiva, jeràrquica i sense pèrdua de senyal.

PCX

És el format emprat pel Paintbrush. Les imatges poden anar des de monocrom fins a color real (24 bits). Aquestes imatges són comprimides fent servir un mètode que ofereix un modest grau de compressió en comparació amb altres compressions de formats. El sistema de codificació que fa servir consisteix bàsicament en comparar els bytes adjacents quan escombra una línia. Si dos bytes són adjacents llavors escriu una bandera amb un comptador juntament amb el byte repetit. Però aquest comptador només pot arribar fins a 15 ja que sinó es poden produir errors. Per això presenta aquest grau baix de compressió comparat amb els altres formats.

PNG

És un format d'imatge en què no es produeix pèrdua de senyal en la compressió. És un mètode molt sofisticat que fa servir 48 o més bits per píxel en les imatges de color i fins a 16 bits de precisió en el mostreig. És un arxiu organitzat en blocs anomenats chunk on s'emmagatzema una certa quantitat de bytes amb el seu corresponent CRC. Tant pot fer servir la paleta de colors com el color real (24 bits). La compressió la pot fer mitjançant el codi de Huffman o l'LC. El codi de Huffman fa servir uns codis de longitud variable enlloc de fer servir un nombre fix de bits per a representar els valors de les components del color del píxel.

TIFF

Aquest format d'emmagatzematge d'imatges té les varietats des de monocrom fins a color real (24 bits). Aquests arxius poden estar en format comprimit o no. El format comprimit té un alt nivell de compressió. L'arxiu està basat en tags. La imatge està definida com una cadena rectangular de píxels. Cada una d'elles s'obté mitjançant un o més mostreigs. Les dades monocromàtiques s'obtenen en un sol mostreig, mentre que els RGB colors necessiten tres mostreigs per píxel.

Altres formats d'imatge que podríem afegir serien el **AVS, BIFF, CGM, CTF, FIG, FITS, PIC, CLP, ICC, IFF, NAP, PCD, PIX, RAS, RLE, SGI, SLB, SLD, TGA, XBM, XWD, EPS, XPM o VIFF.**

2.2- Compressió d'imatges

Des de que les imatges típiques necessiten més d'un megabyte per al seu emmagatzematge, molts formats d'arxiu d'imatge han incorporat tècniques de compressió. El que fan aquestes tècniques de compressió és buscar la manera per tal de tenir la mateixa o gairebé la mateixa imatge o una representació seva que ocupi menys espai que l'original.

Entre aquestes tècniques en podem destacar:

RLE- Run Length Encoding

Fa que píxels consecutius amb el mateix valor siguin codificats només donant la longitud de la cadena de píxels iguals i el seu valor.

LZ Encoding

El compressor manté un diccionari que conté seqüències de valors de píxels que van apareixent en la imatge. La cadena comprimida conté els codis que representen les entrades en el diccionari.

Codi de Huffman

Més que fer servir un nombre fix de bits per a representar els valors dels components, el que fa és usar un codi de longitud variable.

DCT- Transformada Discreta del Cosinus

Blocs de píxels són representats usant la funció cosinus a diferents freqüències. Les altes freqüències, les quals generalment contribueixen amb poca informació en la imatge, són descartades.

La eficiència d'una tècnica de compressió depèn molt sovint del tipus de dades. Els diferents tipus de format d'imatge poden usar diferents tipus de compressió. El format BMP només usa la compressió RLE, el format GIF la LZ, però en canvi el format PNG tant pot usar la compressió LZ com la de Huffman, i el JPEG pot usar compressió RLE, Huffman o DCT.

2.3 El format d'imatge JPEG

2.3.1-Compressió JPEG enfront altres formats comprimits

El JPEG estàndard és bastant complex perquè més que definir un format d'imatge el que fa és definir un nombre de tècniques de compressió d'imatges relacionades. Aquest format obté una gran compressió si ho comparem amb altres formats que usen mapes de bits.

Per exemple, si comparem els resultats d'una compressió d'imatge d'una fotografia podem veure que si l'hem reduït al 10 % amb el format JPEG, la resta de formats no arriben al mateix grau de reducció. Per exemple amb PNG s'arribaria a reduir-se fins al 40 %, amb el format GIF al 70 % mentre que amb el BMP-RLE la reducció pràcticament és inapreciable.

Tanmateix, si enlloc d'una fotografia el que tenim és un dibuix amb poc colors aleshores totes les tècniques de compressió ens donen uns resultats molt similars.
Així doncs, tenim que el format JPEG no és útil per a totes les aplicacions, ja que és un mètode en el que es produeix pèrdua d'informació. Així, si tenim que la compressió dóna un resultat similar amb diferents formats de compressió, procurarem agafar els formats que no ens introdueixin cap pèrdua d'informació. Per tant no farem servir JPEG per a compressió de text i dibuixos, i sí en canvi quan calgui comprimir fotografies. Aquesta pèrdua d'informació, també l'hem de tenir en compte quan s'han de fer repetidament emmagatzematges intermedis en el cas de edicions de la imatge.

2.3.2- Avantatges i desavantatges de la compressió amb o sense pèrdua d'informació.

Una compressió sense pèrdua és la que agafa una imatge, la comprimeix usant una tècnica determinada, i després l'expandeix donant com a resultat una imatge bit per bit idèntica a l'original. En canvi una tècnica de compressió amb pèrdua el que ens dóna després de l'expansió és una imatge molt similar a l'original però no exactament la mateixa. El que passa és que aquesta imatge resultant malgrat no ser l'original, a l'ull humà li resulta francament difícil distingir colors gairebé idèntics i troba la imatge indistingible de l'original.

Per altra banda, tenim que les tècniques de compressió amb pèrdua d'informació permeten un alt grau de compressió respecte al que donen les que no tenen pèrdua. Així doncs, petites pèrdues de dades són acceptables a canvi d'un alt increment en el grau de compressió de la imatge.

2.3.3-Tipus de compressió JPEG

L'original JPEG estàndard defineix quatre modes de compressió: jeràrquic, progressiu, seqüencial i sense pèrdua de senyal.

2.3.3.1- Seqüencial

Aquest mètode el que fa és codificar seqüencialment des del primer píxel de la línia superior fins al darrer de la línia inferior de la imatge. Cada component del color és codificada només amb un mostreig senzill, que dóna un bloc de dades comprimides que conté un sol pas a través de la imatge per a un o més components. A diferència de la majoria de formats en què la totalitat dels píxels comprimits són emmagatzemats en una regió contigua de l'arxiu, en aquest cas tenim que cada passada a través de la imatge és emmagatzemada en un bloc diferent de dades.

Aquest mètode té dues alternatives per a codificar la imatge: usant el codi de Huffman o mitjançant un codi aritmètic. Les dues alternatives tenen la possibilitat de codificar amb 8 o 12 bits per píxel. El JPEG estàndard també defineix una altra codificació seqüencial que usa codi de Huffman: és el procés *baseline*. Un decodificador que processa trames esteses pot ser capaç de processar trames baseline de forma transparent .

2.3.3.2 – Progressiu

Les imatges JPEG progressives tenen els components codificats en múltiples escombrats. Les dades comprimides per cada un dels components són obtingudes amb un mínim de 2 i com a molt 896 escombrats, encara que gairebé sempre s'usa el valor més baix d'aquest rang. L'escombrat inicial crea una basta versió de la imatge mentre que subsegüents escombrats van refinant. Aquest tipus de compressió permet anar veient el que es va decodificant, per tant és molt útil quan es descarreguen imatges d'una xarxa o quan es busca en una Web, ja que permet fer-se una idea de la imatge que conté l'arxiu amb només unes poques dades que han estat transmeses.

El principal handicap que té és la dificultat d'implementar molt superior a la compressió seqüencial. I si la imatge és visualitzada mentre s'està descarregant, aleshores necessita molt més processament. No és gaire encertat fer-ho servir quan la potència relativa de processament és superior a la velocitat de transmissió de la imatge. Més o menys les imatges en aquest format ocupen el mateix que amb el format de compressió seqüencial. Aquest format a l'igual que el seqüencial té les alternatives d'usar codi Huffman o codi aritmètic, en ambdós casos amb mostres de 8 o 12 bits. Rarament s'usa, però si és el cas, aleshores lo més comú és el codi Huffman amb mostres de 8 bits.

2.3.3.3- Jeràrquic

Aquest mètode de compressió fa que les imatges es redueixin a un nombre d'imatges més petites anomenades trames. Cada trama és una col·lecció d'un o més escombrats. En aquest mode, la primera trama crea una versió de la imatge a baixa resolució . La resta de trames el que fan és definir de nou la imatge incrementant la seva resolució.

Es podria dir que aquest mode és millor que el progressiu quan es transmet a baixa velocitat. Si es desitja una baixa resolució d'imatge, aleshores no totes les trames són necessàries per aconseguir el resultat desitjat.

Té el desavantatge de ser un mode molt complex. Clarament requereix molt més procés que els altres modes i fa servir múltiples trames incrementant la quantitat de dades que han de ser transmeses. No es fa servir gairebé mai.

2.3.3.4- Sense pèrdua

Aquest mode de compressió preserva sempre l'exactitud de la imatge original, malauradament això implica una menor compressió que la resta de modes. Per a la majoria d'aplicacions aquest mode no comprimeix tant bé com els que ja hi són disponibles, fent que aquest mode no sigui usat. Sobretot ara que ha aparegut un nou mètode anomenat JPEG-LS per al mateix propòsit que l'ha fet totalment obsolet.

2.3.4- Anàlisi detallat del JPEG usat en el TFC.

2.3.4.1- Nocions bàsiques.

2.3.4.1.1- Ordenació de bytes

Els nombres seran emmagatzemats amb el byte més significant en primer lloc, és a dir, usant el format *big-endian*. Les cadenes de bits seran codificades dintre de bytes començant amb els bits de més pes.

2.3.4.1.2 – Freqüència de mostreig

Per convertir una fotografia en una imatge digital cal agafar mostres de la imatge i mesurar els valors dels components a intervals regulars. Quant més freqüentment se'n agafin de mostres, millor serà l'aproximació a la imatge original.

JPEG permet als components individuals ser mostrejats a diferents freqüències. Això permet que les imatges siguin comprimides mentre anem variant la quantitat d'informació assignada a cada component. Si agafem el mode de color YCbCr, tenim que el component Y és el més important mentre que la resta no en són

tant. Aleshores si reduïm la quantitat d'informació inclosa en la imatge procedent de Cb i de Cr, tindríem una manera de reduir la mesura de la imatge comprimida.

El procés de reduir el nombre de dades per component s'anomena *down-sampling*. Un codificador pot implementar a la baixa el mostreig només codificant cada segon, tercer o quart píxel o bé agafant un valor mitjà. El procés contrari s'anomena *up-sampling*. El mètode més simple d'aquest mostreig és repetir un mateix valor a través de múltiples píxels. A vegades per a reduir els efectes de bloc es fa servir filtratge.

2.3.4.1.3- Descripció del procés

Els passos que es segueixen per tal d'obtenir un arxiu JPEG partint de la imatge original són:

- Mostreig

En aquesta etapa les dades dels píxels són convertits del format RGB al YCbCr i seguidament són mostrejats.

- Transformada discreta del cosinus (DCT)

En aquesta etapa les imatges JPEG són comprimides en blocs de mida 8 de costat anomenats unitats de dades. La DCT converteix els valors de les unitats de dades en una suma de funcions cosinus.

- Quantització

Aquesta fase elimina els coeficients obtinguts amb la DCT que no són essencials per a fer una recreació molt aproximada de l'original. Aquest procés és el que fa que aquest tipus de compressió sigui amb pèrdua.

- Codificació Huffman

Aquí es codifiquen els coeficients quantitzats mentre es van eliminant cadenes de valor zero. En aquesta fase si utilitzéssim una codificació aritmètica aleshores estaríem fent servir l'altre mètode de la compressió JPEG.

2.3.4.1.4- Escombrat a un sol component o a més d'un

Un escombrat pot contenir dades per a un o més components. Un escombrat amb un sol component s'anomena *noninterleaved*. L'escombrat es fa de dalt a baix i d'esquerra a dreta, i les unitats de dades són codificades en una fila.

En el cas de que hi hagi més d'un component, anomenat *interleaved*, el resultat de l'escombrat es codifica en grups d'unitats de dades anomenats MCU. Una

freqüència de mostreig de component vertical especifica el nombre de fileres d'unitats de dades que formen part d'un MCU i la freqüència horitzontal dóna el nombre de columnes.

Els escombrats sempre són codificats amb MCU complets. En cas de que la imatge no sigui múltiple d'aquesta mida, s'inseriran valors de píxels extra en les dades de la imatge per tal de que les dades comprimides sempre tinguin MCU complets.

2.3.4.2- Codi de Huffman

2.3.4.2.1- Idea general

El codi Huffman no s'aplica sobre els píxels sinó sobre els coeficients DCT transformats. Per això, es parlarà de símbols d'una seqüència enlloc de píxels d'una imatge.

El que fa aquest codi és mirar la freqüència amb què un símbol surt en una seqüència i assignar-li un valor binari en funció d'aquesta freqüència d'aparició. El valor binari assignat és de longitud variable de manera que als símbols més usats se'ls assignaran els valors de longitud més petita, mentre que els menys usats seran els de major longitud.

Agafem una seqüència amb un nombre determinats de símbols, cadascun d'ells té un color determinat (cada símbol té un valor entre 0 i 255 associat a la tonalitat del color). Aleshores fem un llistats dels colors que apareixen juntament amb la seva freqüència. Després el que fa el codi és fer un arbre binari on les fulles seran els colors menys freqüents i a mesura que ens anem acostant a l'arrel apareixen els colors que apareixen amb més freqüència a la seqüència.

Agafem cada dos colors de la mateixa freqüència i els unim a una branca a la que s'afegirà l'altre color de major freqüència. Pot haver-hi més d'una branca del mateix valor, creant un arbre de diferents nivells associats a la freqüència dels color que hi són ficats. A cada rama de l'arbre se li assigna un 1 o 0 (és binari). I així tenim que cada color, que seria una fulla de l'arbre està format per una cadena de bits que vénen donats pel recorregut que s'ha de fer per a arribar des de l'arrel fins a la fulla en qüestió. Així la fulla que surt de l'arrel vindria definit per un sol bit (0 o 1), que definiria el color més usat, mentre que el color menys usat estaria definit per una llarga cadena de uns i zeros.

Per tal d'optimitzar aquest sistema es pot fer que els arbres s'equilibrin sempre tenint en compte la freqüència dels colors, amb la qual cosa s'obtenen cadenes d'uns i zeros de menor longitud.

Hi ha una altra manera de generar codis de Huffman diferent del sistema d'arbres i seria generant una longitud de codi per cada símbol i d'aquests generar el codi final.

Agafant la llista de símbols o colors amb la seva freqüència es van agrupant els de la mateixa freqüència sumant els seus valors, així podem afegir-hi l'altre símbol amb el mateix valor que l'obtingut i així successivament. Així tenim una cadena de símbols als que associem amb el nombre que han format part d'una nova agrupació. Així obtenim el codi de Huffman per a cada un d'ells amb tants bits com d'agrupacions n'hagin format part i incrementant el comptador del valor donat al símbol anterior que hi tenia el mateix nombre d'agrupacions.

2.3.4.2.2 – Codi de Huffman en JPEG

El JPEG estàndard posa dues restriccions al codi de Huffman. Per una part no permet que el codi de Huffman consisteixi en una cadena de bits tots de valor 1. Per altra part limita la longitud de codi a 16 bits. En el primer cas s'assigna el codi de la cadena d'uns a un valor imitatiu (*dummy*), que serà descartat al final del procés de codificació. Mentre que per al segon la solució és un xic més complicada ja que cal trobar un símbol amb una longitud de codi com a mínim menor que dues vegades el codi a reduir. Una vegada trobat es substitueix el símbol amb una branca i en ella es col·loquen els codis de major longitud.

El mètode per a decodificar els valors del codi Huffman consisteix en anar creant un arbre binari que contingui els valors posats en un determinat ordre segons el seu codi. Es comença a l'arrel de l'arbre i, usant els valors dels bits llegits procedent de la cadena es determina el camí de cadascun dels valors en l'arbre.

Un altre mètode més senzill d'implementar seria usar la llista de valors ordenats pel codi de Huffman en conjunció amb la cadena de l'estructura de dades amb un element per longitud de codi de Huffman. Cada estructura contindria el mínim i el màxim codi de Huffman per a una longitud donada i l'índex al primer valor amb aquesta longitud d'aquest codi dins la cadena de valors ordenats.

Resumint podríem dir que els codis de Huffman comprimeixen assignant codis de longitud més curta als caràcters més freqüents.

2.3.5 – Creació d'un arxiu JPEG seqüencial

A diferència del procés de descodificació, en el qual ja tenim especificats els diferents paràmetres, en el procés de codificació d'un arxiu JPEG hi intervenen determinats factors que no són especificats. Per especificar aquests factors cal decidir entre vàries possibilitats o mètodes. Per exemple, hem de decidir entre els diferents paràmetres de compressió, el codi de generació Huffman i el tipus d'ordenació dels blocs.

2.3.5.1 – Paràmetres de compressió

En comprimir una imatge usant el mode seqüencial de compressió JPEG, cal triar entre diferents opcions, que són:

- La composició i el nombre de taules de quantització i l'assignació de taules per component.
- El nombre de taules de Huffman, la seva assignació de components, i el mètode per la seva creació.
- El nombre d'escombrats i els components en cadascun d'ells.
- La freqüència del marcadors de reinici.
- Les freqüències de mostreig..
- Si cal crear una imatge a color o amb escala de grisos.
- Si cal usar baseline o trames esteses.

Un codificador por fer aquestes eleccions tant per una imatge base o globalment.

2.3.5.2 Descompressió

En aquest procés es va decodificant la informació de l'arxiu en ordre invers a com s'ha comprimit i al final es recupera la imatge.

3- Fonaments teòrics del treball

3.1- Fonaments de protecció del copyright

En el comerç electrònic pur, on l'article a vendre està en suport electrònic, se'ns presenta el problema de la distribució il·legal: la pirateria.

Per tal de resoldre aquest problema estan apareixent en els últims anys diferents tècniques destinades a marcar els productes (exemples de diferent programari així com companyies que s'hi dediquen els podeu trobar a: http://www.cl.cam.ac.uk/~fapp2/steganography/products.html).

Entre aquests en podem destacar Microsoft DRM que és una aplicació que encripta arxius multimèdia digitals i la seva llicència. El paquet conté una versió de l'arxiu que ha estat encriptat amb una clau de manera que només la persona que ha obtingut la llicència es capaç de fer-ho servir. La llicència ha estat separada del paquet, això permet que el paquet i la llicència es puguin adquirir vàries vegades. Els arxius encriptats poden ser descarregats a l'ordinador del consumidor.

Autokey de Cognicity és un sistema de marcatge del copyright per audio. Copysight d'Intellectual Protocols és un servei basat en java per tal de protegir els drets intel·lectuals davant els pirates. També podem destacar Cryptolope d'IBM, també fet en java.

Com ja hem apuntat anteriorment, la idea és realitzar una sèrie de marques imperceptibles en el producte a vendre de manera que la duplicació del producte implicarà la duplicació de les marques. Si bé la base teòrica de la protecció del copyright és comuna a qualsevol article en format electrònic, de cara a centrar-nos en el nostre treball, només tractarem imatges.

Podem distingir entre dos tipus de marques: el watermarking o "marques d'aigua" i el fingerprinting o "empremptatge". El watermarking insereix un missatge de copyright en la imatge, anàleg al símbol que apareix en les publicacions impreses en paper. El fingerprinting introdueix un mena de número de sèrie, quelcom que identifica al comprador d'aquella imatge concreta. Així doncs, en els esquemes de watermarking, la marca inclosa en la imatge és la mateixa per tots els usuaris, mentre que en els de fingerprinting depèn de la identitat del comprador. La utilització dels dos esquemes també és diferent. Mentre que el watermarking serveix només per provar l'autoria de la imatge, el fingerprinting és més potent i permet identificar el redistribuidor il·legal d'una imatge. Ambdós tècniques requereixen de certes propietats comunes però el fingerprinting, a més també ha de resistir atacs de confabulació de diferents usuaris.

3.2- Processos de marcatge i recuperació de la marca

En qualsevol cas, watermarking o fingerprinting, els processos generals de marcatge i reconstrucció de la marca es poden esquematitzar amb els gràfics de la fig.3.1.

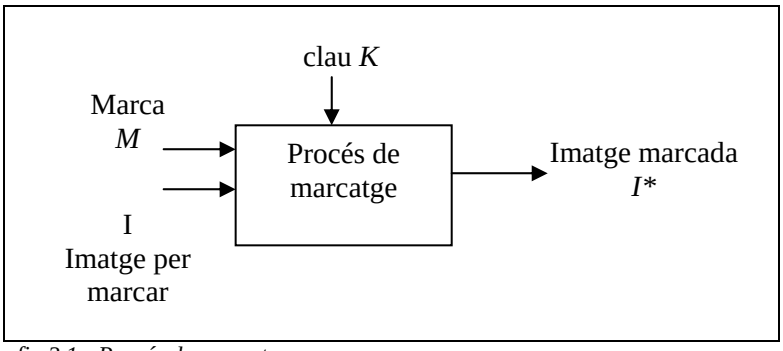

fig.3.1.- Procés de marcatge

Com veiem, en el procés de marcatge l'algorisme té com a entrades una clau secreta *K* (que serveix per distribuir secretament els bits de la marca dins de la imatge), la marca que es vol inserir M, i la imatge que es vol marcar I. Com a sortida, l'algorisme donarà la imatge marcada *I**.

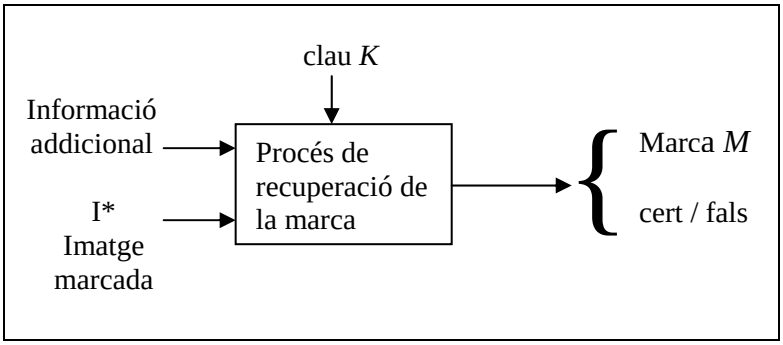

fig.3.2.-Procés de recuperació de la marca

En el procés de recuperació de la marca M, les entrades de l'algorisme seran la imatge marcada i certa informació addicional (que dependrà del tipus d'esquema que fem servir). Com a sortida obtindrem la marca *M* que hi havia en l'objecte marcat o bé (també depenent del tipus d'esquema que fem servir) un valor binari que ens indicarà si la comprovació que fa el procés de recuperació és certa o falsa. Aquest esquema queda il·lustrat a la fig.3.2.

3.3- Propietats dels esquemes de protecció del copyright

3.3.1. Propietats generals

Les propietats que podem demanar a un esquema de protecció del copyright són les següents:

Imperceptibilitat: La marca que realitzem en la imatge ha de ser imperceptible per al comprador. És a dir, la qualitat de la imatge no pot disminuir a causa de la inclusió de la marca.

Robustesa: Aquesta propietat ens mesura la resistència de la marca contra manipulacions o atacs que pot sofrir la imatge. Aquesta propietat és potser la més important ja que l'objectiu final d'un esquema de protecció del copyright és el de mantenir la marca dins la imatge a menys que la imatge resulti tan manipulada que perdi el seu valor com a tal.

La marca introduïda ha de seguir estant retingut dins la imatge malgrat que es facin atacs de distorsió per tal de treure-la o modificar-la. Així doncs, una marca fràgil, feta per un procés de marcatge no gaire bo, provoca que la marca pugui ser eliminada de la imatge, o bé, pot fer-la no identificable, aconseguint els modificadors d'imatges el seu objectiu.

D'atacs se'n poden distingir de dos tipus:
- atacs d'un sol atacant
- atacs de confabulacions de diversos atacants

Els atacs d'un sol atacant són els que pot realitzar un comprador amb la seva còpia de la imatge marcada. Aquest tipus d'atacs es basen en distorsions imperceptibles de la imatge (per veure possibles atacs per a imatges i programari que els realitza es pot trobar més informació a http://www.cl.cam.ac.uk/~fapp2/watermarking/stirmark/).

Els atacs de confabulació són aquells que es realitzen en base al coneixement de més d'una còpia de la imatge. Aquest tipus d'atacs es poden realitzar sobre esquemes de fingerprinting però no tenen sentit en esquemes de watermarking ja que aquests últims marquen totes les imatges amb la mateixa informació i per tant el coneixement de diferents còpies no aporta cap informació extra. En el cas de fingerprinting, com que les marques que s'introdueixen per cada comprador són diferents, comparant diferents còpies es podria revelar la posició d'alguns bits de la marca.

Taxa d'informació: Aquesta propietat mesura la quantitat d'informació que es pot introduir en una imatge sense que afecti la seva qualitat. Es pot mesurar com la longitud en bits de la marca inclosa. De cara a augmentar la robustesa de l'esquema, normalment la informació que s'insereix en cada imatge s'inclou de forma redundant. Aquest fet fa que el valor net de taxa d'informació de l'esquema es redueixi. Caldrà doncs, obtenir un equilibri entre taxa d'informació i robustesa.

Seguretat per al comprador: Ha d'assegurar-se la identificació del propietari o comprador d'una còpia trobada a posteriori, de manera que el comprador ha de poder estar segur, en tot moment, que no serà acusat falsament de redistribució il·legal.

Anonimat: el comprador ha de poder preservar el seu anonimat.

3.4- Classificació dels esquemes de protecció del copyright

Com ja hem dit anteriorment, els esquemes de protecció del copyright és divideixen en dos grans grups: esquemes de watermarking i esquemes de fingerprinting.

3.4.1 Watermarking

Els esquemes de watermarking admeten la següent classificació (representada en la següent fig.3.3) depenent de la informació necessària en el procés de recuperació de la marca.

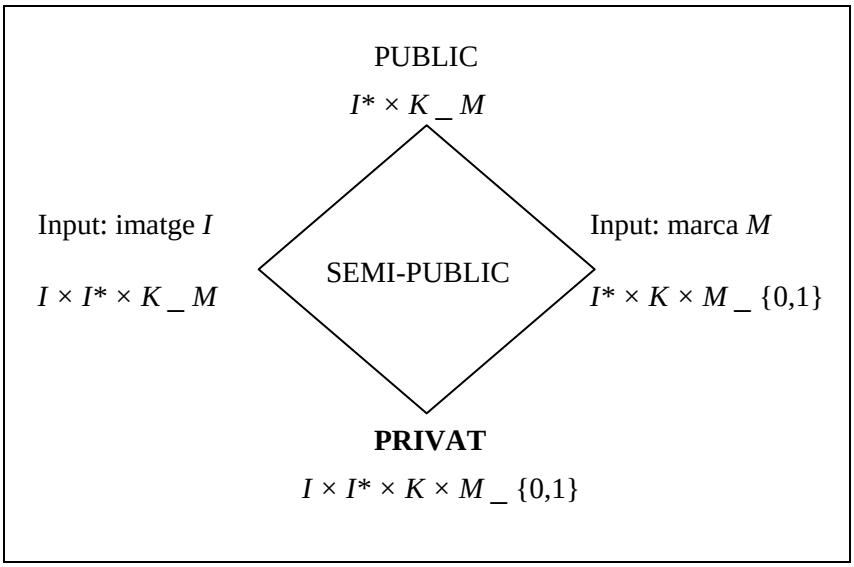

fig.3.3.-Classificació d'esquemes watermarking

Els esquemes de **watermarking privat** són aquells que necessiten més informació per la recuperació de la marca. En concret, es necessita la imatge original I, la imatge marcada *I**, la clau utilitzada per marcar *K* i la marca inclosa M. Com a sortida s'obté cert/fals que determina si és cert que la marca donada com a entrada està continguda en la imatge marcada o si no hi és present.

Els esquemes de **watermarking semi-public** no necessiten tanta informació com els privats. En trobem de dos tipus, els que necessiten la imatge original per recuperar la marca, però no necessiten saber quina marca hi havia inclosa i els que no necessiten la imatge inicial però si la marca. Aquests últims només poden donar com a sortida un argument que determina si la marca donada està inclosa en la imatge marcada o no.

Finalment els esquemes de **watermarking públic** són els que en tenen prou amb la imatge marcada i la clau per tal d'extreure el valor de la marca inclosa. Tot i ser els millors esquemes que pot haver-hi, ja que necessiten el mínim d'informació per recuperar la marca, hi ha poques propostes d'aquest tipus. A més, aquest tipus d'esquemes no poden resistir atacs de retalls en la imatge ja que al no disposar de la imatge original no poden sincronitzar el tros de la imatge marcada per tal de trobar-ne la marca.

3.4.2 Fingerprinting

Pel que fa als esquemes de fingerprinting, se'n pot diferenciar tres tipus: fingerprinting simètric, fingerprinting asimètric i fingerprinting anònim.

3.4.2.1 Fingerprinting simètric

Com ja hem mencionat anteriorment, els esquemes de fingerprinting són esquemes de protecció del copyright que permeten la detecció del culpable de la redistribució il·legal d'una imatge.

Aquesta tècnica es fomenta en la base de què hi ha una part , la venedora, que ven el seu producte al comprador, i no vol que aquest faci còpies del producte venut. Se suposa que el venedor sempre va de bona fe i és el comprador el què farà les il·legalitats. Per això tenim que és el venedor del producte, que posseeix el copyright el que s'encarrega de la introducció de marques en l'article o informació objecte de la transmissió amb la finalitat de poder reconèixer al comprador d'aquest producte amb posterioritat, si aquest s'ha dedicat a distribuir il·legalment les còpies comprades.

Els algorismes que es fan servir són dos: un que s'encarrega del marcatge de les còpies que es distribueixen, que s'anomena marcatge, i altre que permet establir a qui pertany una còpia del producte trobat, que s'anomena identificació.

El marcatge ha de ser segur per al venedor. És a dir, que permeti identificar al comprador de l'original de la còpia pirata. Aquest comprador pot ser passiu o actiu. Serà passiu si només es dedica a distribuir la còpia sense modificar-la i serà actiu si intenta fer atacs per tal d'esborrar les dades que l'identifiquen.

S'ha de garantir que es pot identificar el propietari d'una còpia, encara que un grup de compradors es confabulin per alterar el contingut de l'arxiu. Quan un grup de compradors es confabula, el que fan és comparar les seves respectives còpies marcades per localitzar diferències que els ajuden a localitzar la posició de les marques sent-los així més fàcils d'eliminar.

Així doncs, el fingerprinting simètric o clàssic és aquell que és segur per al venedor sota cert nombre de confabuladors però que no incorpora la propietat de ser segur per al comprador (veure propietats dels esquemes en l'apartat anterior). Aquest fet es deu a que la còpia marcada la coneixen tant el comprador com el venedor (d'aquí la denominació de simètric). A més consisteix en dos algorismes, el de marcatge i el d'identificació, que seran executats només pel venedor.

Això fa que un venedor deshonest pugui redistribuir la imatge que ha venut a un determinat comprador i després acusar-lo injustament. És més, aquest argument podria ser esgrimit davant d'un jutge per un comprador culpable acusat de redistribució.

3.4.2.2 Fingerprinting asimètric

El fingerprinting asimètric [Pfit96] va néixer per tal d'oferir la propietat de seguretat per al comprador. La principal diferència amb els esquemes simètrics és que l'algorisme de marcatge (que abans l'executava el venedor) passa a ser un protocol en el que hi intervenen tant el comprador com el venedor. Aquests esquemes tenen la peculiaritat que el venedor no veu la còpia marcada que al final obté el comprador. D'aquesta manera, com que el comprador és l'únic que la coneix, la posterior localització i identificació de la còpia per part del venedor acusa el comprador de redistribució de manera inequívoca. Aquesta propietat, que podria semblar estranya, s'aconsegueix utilitzant diferents tècniques: el càlcul segur a dues bandes[Pfit96,Domi98a] (on comprador i venedor donen les entrades de la funció i només el venedor obté la sortida: la còpia marcada), la transferència inconscient combinada amb proves de coneixement nul [Domi98c] o l'eina criptogràfica de transferència inconscient compromesa [Domi98b].

Podríem dir que és una modificació de l'esquema del fingerprinting simètric on s'han modificat els protocols de marcatge i identificació, a més d'afegir dos més de nous, que són la generació de claus i la disputa.

Els protocols que formen part d'aquest esquema són:

1. Protocol de generació de claus. Realitza la generació de claus per a l'esquema de firma digital.

2. Protocol de marcatge. Intervenen tant el comprador com el venedor, però només el comprador obté el resultat de la còpia marcada. Mitjançant computació segura a dos bandes s'aconsegueix que el venedor no vegi el resultat a canvi d'obtenir un registre de venda que li permetrà identificar la còpia.

3. Protocol d'identificació. És l'algorisme que executa el venedor sobre una còpia trobada usant els registres de venda obtinguts en el protocol de marcatge. L'objectiu és identificar al distribuïdor, comprador de la còpia marcada que s'ha trobat.

4. Protocol de disputa. Intervenen el venedor i l'àrbitre. El venedor presenta proves a l'àrbitre, l'àrbitre comprova la validesa de la prova, i declara culpable de distribució al comprador posseïdor de la clau.

Aquest esquema esdevé segur per al comprador, atès que no existeix cap mètode factible amb el que un venedor deshonest pugui inculpar a un comprador honest.

3.4.2.3 Fingerprinting anònim

El fingerprinting anònim [Pfit97] es pot veure com una analogia als esquemes de pagament anònim, on el comprador no ha de revelar la seva identitat. En el cas de fingerprinting, a diferència dels esquemes de pagaments anònims no-revocables, la identitat del comprador ha de ser emmagatzemada d'alguna manera per tal de poder obtenir-la en cas que es demostri que és culpable de redistribució. Així doncs, la majoria d'esquemes anònims existents és basen en la utilització de pseudònims controlats per una autoritat de registre que és la que controlaria la correlació de les identitats dels compradors i els seus pseudònims.

L'anonimat s'aconsegueix introduint l'ús del pseudònims per als compradors. Aquest fet requereix la introducció d'una nova part que intervé en l'esquema: el centre registrador. Aquesta entitat coneix a quin comprador pertany cada pseudònim.

Els protocols corresponent a aquest esquema són:

1. Protocol de registre. Se realitza entre el comprador i el centre registrador. El centre registrador firma amb la seva clau privada un certificat cert on fa constar que coneix la identitat real del pseudònim i retorna el certificat al comprador

2. Protocol de marcatge. Es realitza entre el venedor i el comprador sota el pseudònim. El comprador ha de firmar amb el seu pseudònim un identificador de la compra.

3. Protocol d'identificació. El venedor extreu de la còpia la informació que el comprador va introduir en el procés de marcatge.

4. Protocol de judici. L'àrbitre verifica la firma del comprador acusat en què aquest es fa responsable del pseudònim.

Aquest esquema permet mantenir l'anonimat del comprador, de tal manera que el comprador honest no podrà ser identificat ni les seves compres podran ser seguides si no és que el centre registrador actuï deshonestament amb cooperació amb el venedor.

Per tal de trobar protocols més eficients en l'ús de la computació segura a dos bandes apareix l'esquema de fingerprinting anònim d'en Josep Domingo, que proposa modificacions en els protocols de registre, marcatge i d'identificació.

4. Esquema de watermarking implementat

4.1. Algorisme d'inserció de marca

Per implementar la funcionalitat de marcatge, el programa ha de rebre com a entrada la imatge que volem marcar, la marca que hi volem introduir i una clau secreta. Com a sortida ens donarà la imatge marcada. També caldrà rebre com a entrada un percentatge de compressió de la imatge, la funció d'aquest s'explica seguidament.

Aquest algorisme agafa una imatge i insereix la informació donada com a entrada dins d'aquesta imatge. Així doncs s'ha de decidir on introduirem aquesta informació i de quina manera ho farem. En un principi, la informació podria ser una cadena de caràcters com podrien ser un codi de números i lletres o bé el nom complet del comprador, però s'ha preferit treballar amb una cadena d'uns i zeros. Així doncs, si ens trobem en el cas d'haver-hi de treballar amb caràcters només caldrà transformar aquests en el seu codi binari per tal de fer servir aquest algorisme.

Per tal de determinar quins píxels de la imatge s'han de modificar i quant s'han de modificar utilitzarem la compressió JPEG. El procés de marcatge haurà de demanar com a entrada un valor q que representarà un percentatge de qualitat de compressió usant l'algorisme JPEG. Obtingut aquest valor, l'algorisme comprimirà al q % la imatge original a marcar I, obtenint-ne I'. La diferència entre aquestes dues imatges I i I' determinarà els píxels a modificar de la següent manera:

- només es marcaran els píxels que hagin estat modificats per la compressió JPEG, és a dir, si definim $d_j=|i_j-i'_j|$ només marcarem els píxels i_j tals que $d_j \neq 0$ (on i_j són els píxels de I i i'_j són els d'I')
- l'increment o decrement dels píxels que marquem és exactament el valor d_j definit anteriorment.

Si ens hi fixem, la imatge comprimida I' només serveix per a veure quins píxels s'han de modificar i quant, ja que les modificacions per obtenir la imatge marcada es fan a la imatge original. Quan ens referim a píxels hem de pensar que ens referim a una imatge a escala de grisos. Si la imatge fos en color aleshores treballem amb la cadena de components que formen la imatge, tres components per cada píxel de la imatge que s'analitzen de forma independent. Podem marcar només un dels components d'un píxel i la resta d'aquest no. Podríem pensar en treballar amb els tres components de cada píxel que fossin modificats alhora per la compressió però això només faria que complicar l'algorisme sense aportar cap millora apreciable.

Una vegada que ja hem determinat els píxels sobre els que introduirem la marca cal buscar la manera de xifrar aquesta informació abans d'afegir-la a la imatge. Així doncs, utilitzarem una seqüència pseudoaleatòria K per xifrar la informació que incloem en la imatge. Per tal que la seqüència sigui segura haurà de ser un valor aleatori. La longitud d'aquesta seqüència serà la mateixa que la de la informació que incloem E. Podríem pensar que fos una seqüència de caràcters com pot passar en el cas d'una password, però això trauria seguretat al xifrat. Per tant el que rep com a entrada l'algorisme és una llavor que fa de clau, aquesta inicialitza el generador pseudoaleatori amb que després crea la seqüència K.

Així la marca final *M* que s'inserirà en la imatge serà la suma XOR bit a bit de la informació que volem incloure *E* i la seqüència *K*, és a dir:

$m_j = e_j \oplus k_j$ on m_j representen els bits d'*M*
e_j representen els bits d'*E*
k_j representen els bits d'*K*

Una vegada tenim la marca ja xifrada i els píxels sobre els que inserim aquesta ja es pot procedir a fer el procés de marcatge de la imatge.

El procés de marcatge es realitza modificant sensiblement la imatge, és a dir certs píxels de la mateixa. Donat que la marca que inclourem en la imatge és una cadena de bits (0's i 1's), per tal de codificar-los en la imatge prendrem la següent convenció:

- si el píxel de la imatge marcada és superior al píxel de la imatge original voldrà dir que hi ha un 1 codificat.
- si el píxel de la imatge marcada és inferior al píxel de la imatge original voldrà dir que hi ha un 0 codificat.

Així doncs, anem buscant seqüencialment els bits de la imatge que havíem marcat i si tenim un bit 1 li afegim al píxel de la imatge el valor d_j definit anteriorment o li restem aquest valor en cas de que el bit sigui el 0. Sempre tenim en compte el que no es sobrepassin els marges, la qual cosa influiria en la qualitat de la imatge. Així si tenim un bit 1 i la suma dels valors sobrepassa el valor 255, aleshores considerarem que el píxel té aquest valor màxim, i quan és un bit 0 i la resta dóna un valor inferior a zero, assignarem aquest valor zero al píxel. Aquestes modificacions es tindran en compte a l'hora de recuperar la marca.

Si només afegim una vegada la marca ens trobem que fàcilment es podria malmetre, per tant caldria anar-la ficant en diferents píxels de tota la imatge per tal de què si alguna de les marques es deteriora les altres puguin servir per extreure posteriorment el valor de la marca.

És per això que, alhora d'inserir la marca dins la imatge, per tal de què l'esquema tingui un elevat grau de robustesa, replicarem la marca tantes vegades com sigui possible. És a dir, si *M* és la marca que volem inserir en la imatge, l'anirem repetint fins a obtenir una cadena *M'* amb tants bits com píxels diferents hi ha entre les dues imatges, és a dir, tants bits com $di \neq 0$.

4.2. Algorisme de recuperació de marca

La funcionalitat d'extracció de la marca necessitarà com a entrades:

- la imatge marcada
- la imatge original sense marcar
- la clau secreta
- el paràmetre q

A més d'això caldrà afegir-li la longitud de la marca.

Com es pot veure es tracta d'un esquema semi-públic segons la classificació feta al capítol 3. Com a sortida el procés ha de donar la marca que ha trobat en la imatge o bé un codi d'error en cas que no s'hagi trobat cap marca.

El procediment és el següent:

Primer de tot determinarem en quins píxels de la imatge hi ha bit de la marca enterrats. Això es farà igual que en el procés de marcatge, és a dir, l'algorisme comprimirà al *q* % la imatge original *I*, obtenint *I'*. La diferència entre aquestes dues imatges *I* i *I'* determina els píxels on hi ha bits de la marca (aquells on $d_j = |i_j - i'_j| \neq 0$). En aquests píxels, determinarem si el bit inclòs és un 0 o un 1 mirant si el valor del píxel de la imatge marcada és menor o major que el de la imatge original respectivament. En el cas que el valor del píxel de la imatge marcada fos 255 ho consideraríem com un bit 1 i si és un 0 es considerarà un bit 0, degut al mecanisme fet servir en la inserció de marca. Per tal d'assegurar-nos de que la marca és la correcta davant modificacions guardarem totes les cadenes introduïdes i les compararem entre elles. La marca no sempre es recupera del tot. Això és degut a què l'algorisme no és 100% robust i hi ha casos en què alguns bits es corrompeixen degut a què la imatge pateix pèrdues en guardar-se en JPEG.

El mètode que es fa servir per tal d'agafar els bits correctes és usar un comptador de bits 1 i bits 0 per cada una de les posicions de la marca. El comptador que tingui un valor superior serà el que decidirà si en aquesta determinada posició tenim un bit 1 o un bit 0. En cas de què hi hagi igualtat s'agafarà un zero.

Així recuperem *M'* i amb el valor de la seqüència *K* podrem extreure la informació *E*. El mètode serà el mateix que el que es fa servir en el cas d'inserció de marca mitjançant la funció XOR.

L'algorisme finalment ens donarà la informació que tenia la imatge marcada.

4.3 Robustesa de l'esquema contra atacs de compressió JPEG

Una vegada que tenim l'algorisme dissenyat caldrà comprovar quina robustesa té respecte a determinats tipus d'atacs. En aquest cas es comprovarà la robustesa de l'algorisme respecte als atacs per compressió JPEG.

Per tal de fer aquest estudi s'agafa una imatge i es fa el marcatge amb una marca d'una determinada longitud. Aquesta haurà de ser prou llarga, amb una llargària d'uns 25 a 30 bits.

Una vegada que es recupera la marca es comprova quin percentatge de bits es van recuperant després dels atacs amb compressió d'imatge.

4.3.1 Experiments que s'han realitzat

Per tal de fer les proves agafo la imatge marcada i la vaig guardant amb diferents factors de qualitat de compressió. Es comença amb la imatge que ens dóna com a resultat d'aplicar un factor de qualitat de 95 i així successivament es van obtenint les imatges resultants fins arribar a uns nivells de factor de qualitat molt baixos, com poden ser 15 o 20.

Per tal de treballar més còmodament s'ha implementat un petit programa que fa servir les llibreries JPEG i que obre la imatge marcada i mitjançant un bucle va guardant-la en imatges amb diferents factors de qualitat. Així doncs, com a resultat de l'execució d'aquest programa, es genera un conjunt d'imatges comprimides amb un factor de qualitat diferent.

Una vegada que tenim aquestes imatges fem la recuperació de la marca per cadascuna d'elles i comprovem quin percentatge d'errors s'ha produït.

També és interessant comprovar com poden influir factors com la llavor o les diferents longituds de marca en el percentatge d'errors produïts.

S'ha estudiat també com afecta el tipus d'imatge a l'hora de fer aquestes proves. Primer de tot s'estudien les imatges en color. Per començar s'agafa una imatge podríem dir estàndard, com pot ser lena.jpg. Després s'estudia una imatge amb molts pocs colors, com és la de crater.jpg, en què la majoria de colors són tonalitats molt fosques i gairebé es podria considerar en blanc i negre sense ser-ho. I finalment tenim la imatge cotxe.jpg en què gran part de la imatge correspon a un mateix color, en aquest cas el groc. S'ha buscat una imatge en que hi han pocs colors, molt similars i amb grans extensions de la imatge amb el mateix color.

Després s'ha estudiat la robustesa en una imatge en blanc i negre. Per fer això s'ha transformat la imatge lena.jpg de color a escala de grisos.

4.3.2 Resultats de les proves

Primera prova

He fet servir:
- la imatge lena.jpg
- una marca de 25 bits (1111101010001101100010101)
- una llavor amb valor 52

he estudiat els casos de diferents qualitats de compressió q=100,75 i 50 en els que he anat variant el factor de qualitat de la imatge marcada.

Els resultats els tenim a la taula 4.1.

Taula 4.1. Resultats de la primera prova

nombre de bits erronis en la cadena resultant de 25 bits			
qualitat de la imatge marcada (%)	q=100	q=75	q=50
100	0	1	1
95	3	6	6
90	5	5	6
85	4	5	6
80	6	9	8
75	8	10	10
70	7	10	11
65	9	11	11
60	8	11	11
55	7	11	11
50	6	9	10
45	10	11	11
40	7	11	11
35	8	11	11
30	9	10	11
25	12	11	11
20	12	11	11
15	12	11	12
10	14	14	14

Degut al elevat nombre d’errors he mirat de buscar altres tipus d’imatge per a veure com responia l’algorisme.

Segona prova

En un primer moment he pensat en agafar una imatge amb una coloració molt similar, pràcticament en blanc i negre i he agafat l'arxiu crater.jpeg.

Els resultats de la segona prova s'indiquen a la taula 4.2.

Taula 4.2.- Resultats de la segona prova

nombre de bits erronis en la cadena resultant de 25 bits			
qualitat de la imatge marcada (%)	q=100	q=75	q=50
100	0	1	2
95	5	5	6
75	10	11	6
50	6	5	4
30	8	9	9
10	7	10	11

D'aquests resultats tampoc es pot treure cap conclusió, poden semblar millors que en el cas anterior però tampoc són satisfactoris.

Tercera prova

Així doncs, he pensat buscar una imatge amb pocs colors i que gran part de la imatge tingui el mateix color. Aquest és el cas de la imatge cotxe.jpg que gairebé tot és groc. Els resultat d'aquesta prova són a la taula 4.3.

Taula 4.3.- Resultats de la tercera prova

nombre de bits erronis en la cadena resultant de 25 bits			
qualitat de la imatge marcada (%)	q=100	q=75	q=50
100	0	0	0
95	3	5	6
75	10	15	8
50	11	9	11
30	13	11	11
10	11	11	14

D'aquesta prova només puc extreure que quan hi ha pocs elements d'un mateix color, aleshores no influeix el grau de compressió de l'arxiu (q) en la creació d'errors en la recuperació de la informació en la imatge marcada sense atacs.

Quarta prova

Tornant a la imatge lena.jpg si agafem el quadre de bits erronis ens trobem que hi ha moltes caselles amb valor 11. Realment en la majoria d'aquests casos els valors obtinguts de la cadena resultant són els mateixos, i en la pràctica totalitat l'últim bit 1 es transforma en 0. Per tal de comprovar si això ocorre sempre faig una última prova:

- agafo la marca de 30 bit 111110101000110110001010111111
- com a llavor fico el valor 452
- la qualitat de compressió serà q=100

Al comprimir la imatge marcada al 95% obtinc 3 errors, mentre que al 75% tinc 9, al 50 % 11 i al 10% 14.

No hi ha millora significativa, a més es comprova que en aquest cas l'últim bit 1 no es modifica com passava en el cas anterior. El que sí que és cert és que el percentatge d'errors ha millorat, ja que malgrat tenir els mateixos errors, tenim una cadena amb més bits que abans.

El que sí que es manté és la modificació dels cinc primers bits de la cadena que de ser 11111 passen a 10110 en la majoria de casos en què els errors són superiors a 5, excepte les compressions inferiors al 20 %, i casos puntuals. Això és així en els tres tipus d'imatge provades i els factors de qualitat q.

Cinquena prova

Si es treballa amb una imatge a escala de grisos el resultat és diferent. Agafaré la mateixa imatge lena.jpg transformada en una imatge a escala de grisos i agafo els mateixos paràmetres que en el cas anterior:

- una marca de 25 bits (1111101010001101100010101)
- una llavor amb valor 52

Els resultats obtinguts a la cinquena prova són a la taula 4.4.

Taula 4.4.- Resultats de la cinquena prova

nombre de bits erronis en la cadena resultant de 25 bits			
qualitat de la imatge marcada (%)	q=100	q=75	q=50
100	0	0	0
95	0	0	0
90	0	0	1
85	0	0	3
80	1	0	5
75	1	2	4
70	2	2	5
65	2	2	5
60	5	1	5
55	0	0	5
50	0	0	9
45	0	0	5
40	11	8	8
35	11	7	7
30	12	7	10
25	4	3	9
20	15	13	12

En aquesta prova es pot comprovar que aquest algorisme està indicat per tal de marcar imatges a escala de grisos. La marca gairebé es pot recuperar en els casos en què es comprimeix la imatge entre el 75 i el 100% malgrat que la qualitat de la imatge marcada sigui atacada per una compressió de fins al 80 %. També es pot indicar que la marca també es recupera si la qualitat de la imatge marcada ha estat atacada als voltants del 50%. En el cas de treballar amb compressió del 50% ja resulta més difícil el recuperar la marca després d'atacs amb qualitat de compressió inferior al 90%.

4.4 Detalls d'implementació

4.4.1 Llenguatge de programació.

Aquest programa s'ha fet utilitzant el llenguatge C i s'ha treballat en l'entorn linux. Les imatges JPEG han estat manipulades mitjançant unes llibreries per a linux que permeten treballar amb imatges JPEG en C. Aquestes llibreries estan en l'arxiu comprimit *jpegsrc.vb6.tar.gz*. De tots els fitxers que componen aquestes llibreries el programa només en fa servir dos: *libjpeg.a* i *jpeglib.h.*

A més, s'han fet servir dues funcions que faciliten la utilització d'aquesta llibreria des d'un programa. Aquestes funcions estan dins l'arxiu *funcionsjpg.c* i són dues: read_JPEG_file(...) i write_JPEG_file(...). La primera extreu la imatge de l'arxiu JPEG, i la segona agafa una imatge i la transforma a un format d'imatge JPEG. Aquestes funcions han estat modificades per tal de què es pugui treballar amb imatges en color com amb imatges a escala de grisos.

Una vegada creat l'arxiu C dels programes de marcatge i desmarcatge es compilen mitjançant la comanda:

gcc marcatge.c funcionsjpg.o ../jpeg-6b/libjpeg.a –o marcatge

gcc desmarcatge.c funcionsjpg.o ../jpeg-6b/libjpeg.a –o desmarcatge

4.4.2 Sintaxi dels programes

Per tal d'executar el programa cal afegir-li al nom del programa els següents paràmetres:

- la imatge en format JPEG a marcar
- la marca amb la informació a afegir a la imatge(ha de ser una cadena de 1 i 0)
- el factor del grau de compressió de la imatge
- la llavor amb la que obtenim una clau aleatòria

El format de la comanda serà el següent:

marcatge *arxiu.jpg cadena_de_bits factor_de compressió llavor*

Podem posar com a exemple un dels que s'ha fet servir:

marcatge lena.jpg 1111101010001101100010101 75 52

Com a resultat obtenim el nom de l'arxiu marcat en format JPEG

longitud de cadena: 25
arxiu marcat: lena_m.jpg

Per tal de recuperar la marca cal afegir al programa desmarcatge els següents paràmetres:

- la imatge original en format JPEG
- la imatge marcada en format JPEG
- la longitud de la cadena amb la informació afegida
- el factor del grau de compressió de la imatge
- la llavor amb la que obtenim la clau aleatòria

El format de la comanda serà:

desmarcatge *arxiu.jpg arxiu_m.jpg longitud factor_de_compressió llavor*

Si recuperem la imatge de l'arxiu de l'exemple anterior tenim:

desmarcatge lena.jpg lena_m.jpg 25 75 52

i el programa ens dóna com a resultat la cadena de informació.

informació_final : 111101010001101100010101

4.4.3 Detalls de la programació

Hi ha alguns detalls del programa que no s'expliquen als apartats anteriors sobre el marcatge i recuperació de la marca, com poden ser la generació de la clau o el nom de l'arxiu de sortida, i que passem seguidament a comentar.

Per tal de generar la seqüència aleatòria que ens farà de clau es fa servir la funció srand(unsigned seed) que tenim dins stdlib.h . Aquesta funció s'utilitza per tal d'inicialitzar el generador de números aleatoris a partir de la llavor(seed) i mitjançant la utilització de la funció random() es genera una seqüència de números aleatoris que seran els que ens aniran donant els 1 i 0 de la seqüència que farem servir com per xifrar. En el nostre cas si tenim un nombre aleatori parell, aleshores inserim un 0 a la seqüència i si el nombre aleatori és senar l'hi introduïm un 1. Aquesta seqüència tindrà la mateixa longitud que la cadena de informació. Per tal de distingir si un nombre és parell o senar es comprova el resto d'una divisió per dos.

Per tal de donar un nom coherent a l'arxiu JPEG marcat s'ha pensat a afegir '_m' al nombre de l'arxiu original.

Al ser els elements de la cadena imatge del tipus unsigned char cal transformar aquests elements al tipus int per tal de que al fer les operacions de suma o resta, degut a que en casos puntuals poden provocar problemes.

També cal tenir en compte que malgrat que la imatge marcada s'emmagatzema amb un factor de compressió del 100%, una vegada recuperada s'ha pogut comprovar que

la marca no sempre es recupera del tot. Això és degut a que l'algorisme no és 100% robust i hi ha casos en què alguns bits es corrompeixen degut a que la imatge pateix pèrdues en guardar-se en JPEG. Al ser un problema intrínsec de la compressió JPEG l'algorisme de recuperació de marca ha buscat una manera de poder-lo compensar, com a mínim en els arxius que no han sofert atacs.

La manera que s'ha pensat ha estat emmagatzemar totes les cadenes de marques que s'obtenen de l'arxiu marcat, per tal d'anar comptant el nombre de bits 1 , 0 o erronis obtinguts en cada posició. Les comparacions dos a dos d'aquests comptadors ens donaria amb prou validesa quin valor 0 o 1 és el correcte o detectar els errors que fan la cadena resultant no vàlida. Però les proves han demostrat que el problema de compressió JPEG dóna un nombre d'errors massa elevat. Per tant, s'ha buscat altra manera d'obtenir resultats més correctes. La manera ha estat comparar només els comptadors d'uns i zeros. Així doncs, si el comptador d'uns és major que el de zeros aleshores suposem que el bit de la marca és un 1 i que és un 0 en la resta del casos. Aquest mètode dóna com a resultat la recuperació dels bits de la marca en tots els marcatges amb factor de compressió 100% i en les imatges a escala de grisos.

Aquests bons resultats amb les imatges a escales de grisos, ha propiciat la idea de treballar amb grups de 3 píxels, per tal de marcar només les posicions en què els tres valors corresponents a un píxel de color sofrien la mateixa variació deguda a la compressió JPEG. Els resultats de treballar d'aquesta manera no ha estat tant òptima com s'esperava i s'ha decidit no afegir-la a l'algorisme.

En la imatge marcada no hi ha cap indici de que la imatge ha estat marcada. Així doncs, les marques són imperceptibles. Fins i tot després dels atacs, amb diferents graus de compressió es pot comprovar que els píxels marcats són indistingibles dels que no han estat marcats. Això es pot comprovar amb les imatges marcades en format JPEG que s'afegeixen: lena_m.jpg, crater_m.jpg, cotxe_m.jpg i lena_mE.jpg(imatge marcada amb atacs al 80%)

5 Conclusions

En aquest treball s'ha comprovat que si s'introdueix una marca a un arxiu JPEG aquesta es pot recuperar després d'una sèrie d'atacs de compressió. El que s'ha comprovat en les proves realitzades és que l'algorisme de marcatge i recuperació de marca ha de ser estudiat en funció del tipus d'imatge al que es vol inserir la marca i la manera que fa servir el JPEG per tal de comprimir la imatge. En tot sistema de compressió en què es comprimeix a gran escala hi ha una sèrie de pèrdues de qualitat, que potser a cop d'ull no ens resulta apreciable, però que sí que ho és a l'hora de comparar els diferents píxels entre la imatge original i la comprimida. També és interessant estudiar com es combinen els tres colors que formen cada píxel i la manera que aquests es comprimeixen. Per això s'ha fet un estudi de la compressió d'arxius JPEG abans de treballar amb el marcatge i recuperació de la marca inserida.

S'ha treballat a nivell de píxel en imatges a escala de grisos, i s'han obtingut resultats molt vàlids a l'hora de recuperar la marca a diferents graus de compressió de la imatge que ens serviria per senyalar els píxels a marcar. Després s'ha observat que els algorismes utilitzats tenien molta robustesa davant de l'atac per compressió de la imatge marcada fins arribar a un grau en què la imatge ja començava a perdre qualitat.

Per altra banda si es feia servir aquests algorismes amb imatges en color trobem que en alguns casos es pot obtenir algun resultat correcte, però, en general, hi ha un alt grau d'errors produïts a l'hora de recuperar la marca. Després de molts anàlisis podem assegurar que a mesura que la compressió de la imatge que fem servir per obtenir els píxels a marcar és més gran, es produeixen més errors. La qual cosa no passava en imatges a escala de color. Això ho podríem justificar dient que la compressió JPEG treballa a nivell de píxel, però mentre que les imatges a escala de grisos aquests tenen un sol element, les imatges a color en tenen tres, i la compressió les afecta com a conjunt.

6 Bibliografia

[Ande98] R.J. Anderson and F.A.P. Petitcolas, “On The Limits of Steganography” in IEEE Journal of Selected Areas in Communications, 1998, Vol 16, n. 4, pp. 474-481.

[Cox96] I.J. Cox, J. Kilian, T. Leighton and T. Shamoon, “A Secure, Robust Watermark for Multimedia” in Information Hiding'96,LNCS 1174, Springer-Verlag, 1996, pp. 185-206.

[Domi98a] J. Domingo-Ferrer, “Anonymous fingerprinting of electronic information with automatic identification of redistributors”, in Electronics Letters, vol. 34, no. 13, Jun. 1998, pp. 1303-1304.

[Domi98b] J. Domingo, “Anonymous Fingerprinting Based on Committed Oblivious Transfer”, in Public Key Cryptography'99, LNCS 1560, Springer-Verlag, 1999, pp. 43-52.

[Domi98c] J. Domingo and J. Herrera, “Efficient Smart-Card Based Anonymous Fingerprinting”, in CARDIS'98, LNCS 1820, Springer-Verlag, 1999, pp. 231-238.

[Peti98] F.A.P. Petitcolas, R.J. Anderson and M.G. Kuhn, “Attacks on Copyright Marking Systems” in 2nd Workshop on Information Hiding, LNCS 1525, Springer-Verlag, 1998, pp.218-238.

[Pfit96] B. Pfitzmann and M. Schunter, “Asymmetric fingerprinting”, in Advances in Cryptology-EUROCRYPT'96,LNCS 1070, Springer-Verlag, 1996, pp. 84-95.

[Pfit97] B. Pfitzmann and M. Waidner, “Anonymous fingerprinting”, in Advances in Cryptology-EUROCRYPT'97, LNCS 1233, Springer-Verlag, 1997, pp. 88-102.

[Wall91] G. K. Wallace, “The JPEG Still Picture Compression Standard”, Communications of the ACM, Vol 34, n. 4, 1991, pp. 30-44.

John Miano, “Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP”, ACM Press, 1999.

M. Nelson, J.L. Gailly, “The Data Compression Book”, 2a edició, 1995

ANNEX A

ALGORISME DE PROCÉS DE MARCATGE

```
#include <stdio.h>
#include <stdlib.h>
#include "funcionsjpg1.h"

 main(int argc,char *argv[])
{
  unsigned char *imatge,*imatge2,*marcat2;
  unsigned char *info_inici;
  unsigned char clau[10000];
  unsigned char marca_inici[10000];
  unsigned int *marcatge;
  unsigned char *nom;
  unsigned char arxiu_desti[100];
  int longitud;
  int enter_aleatori;
  int maxim;
  int llarg;
  int width,height,rgb,q;
  int i,j;
  int llavor;

  if(argc!=5) /* s'ha afegit un parametre */
  {
    printf("Sintaxi: %s imatge marca factor_qualitat llavor\n",argv[0]); /* en cas que el nombre de
parametres sigui incorrecte, indiquem la sintaxi */
    exit(1);
   }

  nom=argv[1];
  info_inici=argv[2];
  q=atoi(argv[3]);
  read_JPEG_file(argv[1],&imatge,&width,&height,&rgb);
  llavor=atoi(argv[4]);
  maxim = width*height*rgb;
  marcatge=(int *) malloc(maxim*sizeof(int));

  write_JPEG_file("copia.jpg",imatge,width,height,q,rgb);
  read_JPEG_file("copia.jpg",&imatge2,&width,&height,&rgb);

  for(i=0;i<maxim;i++)
  {
   if(imatge[i]==imatge2[i]) marcatge[i]=0;
    else marcatge[i]=1;
  }

  longitud=strlen(info_inici);
  printf("longitud_cadena_informacio : %d\n",longitud);
  srand(llavor); /* generem la clau a partir de la llavor */
```

```
 for(i=0;i<longitud;i++)
 {
   enter_aleatori=random();
   if(enter_aleatori%2==1)
   {
      clau[i]='1';
      if(info_inici[i]=='1') marca_inici[i]='0';
       else marca_inici[i]='1';
   }
   else{
      clau[i]='0';
      if(info_inici[i]=='0') marca_inici[i]='0';
       else marca_inici[i]='1';
    }
 }

 llarg=0;
 for(i=0;i<maxim;i++)
 {
    if (marcatge[i]==1)
     {
        if(llarg==longitud) llarg=0;
        if(marca_inici[llarg]=='1')
         {
           if((int)imatge[i]+(int)abs(imatge[i]-imatge2[i])>255) imatge2[i]=255;
            else imatge2[i]=imatge[i]+abs(imatge[i]-imatge2[i]);
         }
        else
         {
           if((int)imatge[i]-(int)abs(imatge[i]-imatge2[i])<0) imatge2[i]=0;
           else imatge2[i]=imatge[i]-abs(imatge[i]-imatge2[i]);
         }
       llarg++;
     }
 }

 // anomenem l'arxiu marcat amb el mateix nom que l'original afegint-li la m de marcat
 i=0;
 while(i<100)
 {
    if(nom[i]=='.')
      {
        arxiu_desti[i]='_';
        arxiu_desti[i+1]='m';
        for(j=0;j<4;j++){
          arxiu_desti[i+2+j]=nom[i+j];
          }
         i=100;
     }
    else arxiu_desti[i]=nom[i];
     i++;
 }

  write_JPEG_file(arxiu_desti,imatge2,width,height,100,rgb
  printf("arxiu marcat : %s\n",arxiu_desti);

  free(imatge2);
  free(imatge);
}
```

ANNEX B

ALGORISME DE RECUPERACIÓ DE LA MARCA

```
#include <stdio.h>
#include <stdlib.h>
#include "funcionsjpg1.h"

 main(int argc,char *argv[])
{
  unsigned char *imatge,*imatge2,*marcat2;
  unsigned char clau[1000],info_final[1000], marca_final[1000];
  unsigned int *marcatge;
  int llarg, llargada, llavor, longitud_marca;
  int cont1[250],cont0[250],error[250];
  int maxim,error_r, width,height,rgb,q;;
  int marca_r[1000][250];
  int i,j,k,l;

  if(argc!=6)
  {
     printf("Sintaxi: %s imatge_original imatge_marcada longitud_marca factor_qualitat llavor\n", argv[0] );
     exit(1);
   }

  q=atoi(argv[4]);
  llavor=atoi(argv[5]);
  longitud_marca=atoi(argv[3]);
  srand(llavor);

  for(i=0;i<longitud_marca;i++)
    {
      if(random()%2 == 1)   clau[i]='1';
        else clau[i]='0';
    }
  clau[i]='\0';

  read_JPEG_file(argv[1],&imatge,&width,&height,&rgb);
  maxim = width*height*rgb;
  marcatge=(int *) malloc(maxim*sizeof(int));

  write_JPEG_file("copia.jpg",imatge,width,height,q,rgb);
  read_JPEG_file("copia.jpg",&imatge2,&width,&height,&rgb);

  for(i=0;i<maxim;i++)
  {
   if(imatge[i]==imatge2[i]) marcatge[i]=0;
    else marcatge[i]=1;
  }

  k=0;l=0;
  llargada = strlen(clau);
  read_JPEG_file(argv[2],&marcat2,&width,&height,&rgb);
```

```
 for(i=0;i<maxim;i++)
 {
   if(marcatge[i]==1)
    {
     if(k==llargada){  l++;k=0;  }
     if(marcat2[i]>imatge[i]) marca_r[l][k]=1;
       else  if(marcat2[i]<imatge[i]) marca_r[l][k]=0;
         else marca_r[l][k]=5;
     if (marcat2[i]==255) marca_r[l][k]=1;
     if (marcat2[i]==0) marca_r[l][k]=0;
     k++;
    }
 }

 for(i=0;i<llargada;i++) {
 cont1[i]=0;  cont0[i]=0;  error[i]=0;
 }

 for(i=0;i<l;i++) {
   for(j=0;j<llargada;j++) {
     if( marca_r[i][j]==1) cont1[j]++;
      else if (marca_r[i][j]==0) cont0[j]++;
           else error[j]++;
    }
  }

  error_r=0;
  for(i=0;i<llargada;i++){
    if(cont1[i]>cont0[i]) marca_final[i]='1';
        else if(cont0[i]>cont1[i]) marca_final[i]='0';
                 else
                  {
                    marca_final[i]='M';
                  error_r=1;
                  }
  }

 printf("informacio_final: ");
 if (error_r==1) printf("!! Atencio. Cadena amb errors !!!!!\n");
 for(i=0;i<llargada;i++)
 {
   if(clau[i]=='1')
   {
    if(marca_final[i]=='1') info_final[i]='0';
    else info_final[i]='1';
   }
   else{
   if(marca_final[i]=='0') info_final[i]='0';
   else info_final[i]='1';
    }
   printf("%c",info_final[i]);
 }
 printf("\n");

 free(marcat2);
 free(imatge2);
 free(imatge);

}
```

ANNEX C

IMATGES FETES SERVIR AL TREBALL

S'adjunten les imatges fetes servir a les proves per tal de comprovar el funcionament dels algorismes.

La imatge Cotxe.jpg s'ha escollit perquè no hi ha gran varietat de colors (groc i gris principalment i el mateix color ocupa gran extensió dins la imatge.

Aquí mostrem la imatge abans i després del procés de marcatge.

abans del marcatge: *cotxe.jpg*

després del marcatge: *cotxe_m.jpg*

La imatge Crater.jpg s'ha escollit perquè només hi ha tonalitats d'un mateix color, el gris, amb poca superfície que tingui el mateix color. Les zones de gris es van tallant amb vetes de blanc

abans del marcatge: *crater.jpg*

després del marcatge: *crater_m.jpg*

La imatge Lena.jpg ha estat la primera amb la que s'han fet les proves. És una imatge podríem dir estàndard sobre la que s'estudia el watermarking i el fingerprinting. És pot trobar en molts treballs citats en la Bibliografia

abans del marcatge: *Lena.jpg*

després del marcatge: *Lena_m.jpg*

També s'aprofita la mateixa imatge per l'estudi a escala de grisos

abans del marcatge: *Lenab.jpg*

després del marcatge: *Lenab_m.jpg*

Com es pot apreciar en qualsevol de les imatges no es veu cap diferència entre la imatge abans del marcatge i la de després d'introduir-li les marques. Així doncs el marcatge és imperceptible.