Incorporació de noves tècniques de compressió a l'estàndard CCSDS-123

Àlex Bonillo Serra

Resum— Una de les branques de la informàtica que ha augmentat més la seva presència en la societat mundial durant els últims anys és l'àmbit multimèdia i més concretament l'intercanvi massiu d'imatges. Quan es parla d'intercanvi de dades es parla, evidentment, de captar, gestionar i fer possible l'enviament d'aquestes dades. En aquest treball es treballa especialment amb imatges de gran àrees geogràfiques preses des de satèl·lits o avions a gran alçaria, interessants per fer anàlisis meteorològics o estudis de la geografia de la terra; i amb l'estàndard CCSDS-123 que recomana implementacions per crear aplicacions de codificació i tractament d'aquest tipus d'imatges. D'aquesta manera, es veurà el funcionament intern de les recomanacions d'aquest estàndard, sobretot en la fase de codificació i descodificació, on s'afegirà un nou mòdul de codificació aritmètica amb diverses configuracions diferents, per tal de veure de quina manera influeix en el rendiment d'aquest estàndard. Els resultats obtinguts mostren una millora el rendiment, depenent del tipus d'imatges amb que s'aplica el mètode corresponent, però en qualsevol cas es troben mètodes que podrien obrir les portes a noves millores futures.

Paraules clau—CCSDS, CCSDS-123, codificació, descodificació, codificació lossless, codificació Aritmètica, Codificador DumbMQ, taules de probabilitat, QEmpordà.

Abstract—Over the last few years, one of the branches of information technology that has increased its popularity the most around society, is the field of multimedia data and more specifically massive data exchanging. Talking about data exchanging means, obtaining, managing and preparing it to be sent. In this report, the work is done especially over images of wide geographical areas, taken from satellites and aircraft at high altitudes, which are interesting to perform meteorological analysis or studies over earth's geography; and with CCSDS-123 standard, which recommends implementations to create coding and management apps for these kinds of images. Therefore, inner workings of this standard will be seen, especially in coding and decoding phases, where a new arithmetic coding module will be added with different configurations, just to see the way it improves this standard's performance. The results obtained show an improvement in the performance, depending on the type of images in which this method is applied. In any case, new approaches that can help improving this method have been found.

Index Terms—CCSDS, CCSDS-123, coding, decoding, lossless coding, arithmetic coding, DumbMQ coder, probability tables, Qemporda.

1 Introducció

Les imatges digitals són les dades multimèdia que més s'utilitzen, transfereixen i reprodueixen avui en dia. Hi ha molts tipus diferents d'imatges, depenent sobretot dels objectius, del tipus de dispositius que les adquireixen. Aquest treball es centra en la compressió sense pèrdua (lossless) d'imatges per a sistemes d'informació geogràfica, més conegudes com imatges de ''remote sensing''.

A l'hora de captar, tractar i distribuir aquest tipus d'imatges, s'ha de tenir en compte que les condicions no són les mateixes que quan es treballa amb imatges adquirides amb càmeres d'ús personal. Les imatges de remote sensing són capturades amb sensors que es troben en sistemes aerotransportats com avions o satel·lits. Ambós sistemes tenen certes restriccions pel que fa a la transmissió i els dispositius utilitzats per comprimir les imatges.[1][2]

En sistemes instal·lats en avions, les imatges són adquirides i emmagatzemades i un cop l'avió arriba a terra totes les imatges són comprimides en el format desitjat. Per altra banda, en sistemes instal·lats en satèl·lits les imatges són adquirides, comprimides en satèl·lit i enviades a la terra pels canals de comunicació corresponents. En aquest cas, la compressió juga un paper fonamental, ja que els temps de transmissió són limitats degut al moviment orbital dels satèl·lits.

A més, cal tenir en compte que l'equipament instal·lat en els sistemes aerotransportats no pot ser el mateix que el que hi ha a terra, ja que els recursos són limitats. La majoria de satèl·lits que adquireixen les imatges venen equipats amb targetes *Field-Programmable Gate Array*, que no destaquen per la seva complexitat ni capacitat de còmput [3]. Això provoca la necessitat de dissenyar sistemes de compressió simples però molt eficients en rendiment de compressió.

El Consultative Commitee for Space Data Systems (CCSDS) es centra en l'especificació de sistemes de compressió per ser inserits en sistemes aerotransportats. Des de la seva fundació, l'any 1982, s'han definit 3 estàndards: CCSDS-121[4], CCSDS-122[13] i CCSDS-123[5]. Aquest treball es centra en la millora de l'estàndard CCSDS-123. El sistema de compressió del CCSDS-123 està format, principalment, per dues etapes: un predictor, que permet reduir la redundància de les dades a codificar; i el codificador lossless que s'encarrega de codificar aquestes dades predites. Per tal de millorar el rendiment de compressió de l'estàndard CCSDS-123 s'ha substituït l'etapa del codificador de l'estàndard per un codificador aritmètic.

La resta del treball es troba estructurat de la següent manera. La secció 2 descriu els objectius d'aquest treball, la secció 3 fa un breu resum de l'estat de l'art de les tècniques utilitzades en la compressió d'imatges de remote sensing. La secció 4 mostra les tasques realitzades i la metodologia. I, finalment, les seccions 5 i 6 mostren els resultats i les conclusions obtingudes respectivament.

2 OBJECTIUS

Els objectius del treball es marquen, evidentment, tenint en compte les qüestions comentades en l'apartat anterior. Així doncs, l'objectiu principal és el de millorar l'eficiència de l'estàndard CCSDS a l'hora de comprimir les imatges.

Així doncs, el que es proposa en aquest treball és modificar la part més crítica del CCSDS-123, per tal de millorar els rendiments de compressió, obtenint menors mides de fitxers comprimits. Per tal de realitzar aquestes modificacions s'utilitzarà la versió implementada del CCSDS-123 per part del Group of Interactive Coding of Images (GICI) [6]. Aquesta implementació realitzada pel grup d'investigadors de la Universitat Autònoma de Barcelona, s'anomena software QEmpordà i és completament lliure per ser descarregat i provat[6].

Per tal que l'objectiu principal esmentat anteriorment, arribés a bon port, es van definir 3 sub-objectius diferents:

- *Iniciació al QEmpordà:* per tal de dur a terme les tasques corresponents, de forma correcta i que les proves posteriors tinguessin èxit, s'havia de fer un estudi i s'havia de passar un període de prova per comprendre totalment el funcionament del QEmpordà.

- *Implementació del codificador Aritmètic:* Un cop s'ha entès totalment el funcionament de l'aplicació QEmpordà, serà l'hora d'iniciar la implementació dels mòduls codificadors que ens han de permetre fer noves proves i cercar millores en els resultats. El mòdul que afegirem al CCSDS-123 serà un codificador aritmètic no adaptatiu. Aquest codificador es troba actualment implementat pels investigadors del GICI en un paquet conegut com DumbMQ. L'important en aquesta tasca és aconseguir enllaçar l'aplicació QEmpordà amb el mòdul esmentat i que ens permeti realitzar un procés de compressió lossless, és a dir, la imatge final després de passar pel mòdul codificador i descodificador, ha de ser exactament igual a la imatge original.

- *Cerca d'optimitzacions i obtenció de resultats:* Com a últim punt del projecte i un cop havent enllaçat del mòdul codificador DumbMQ amb l'aplicació QEmpordà, s'hauran de revisar els resultats obtinguts i provar noves configuracions pel mòdul implementat. Presumiblement aquestes noves configuracions havien de millorar els resultats obtinguts en la part anterior del projecte, en que només interessava provar que el nou mòdul funcionés correctament. Un cop trobades i implementades les noves millores, serà important també analitzar i presentar els resultats de forma correcta.

3 ESTAT DE L'ART

Pel que fa a les tecnologies que apareixen en aquest projecte, sembla interessant contextualitzar-ne les més importants: estàndards de compressió d'imatges establerts pel CCSDS i treballs actuals dels codificadors aritmètics i els seus tipus.

Així el *Consultative Commitee for Space Data Systems* es va fundar l'any 1982, com un fòrum de discussió sobre problemes i recomanacions en la gestió i la creació de sistemes de dades espacials[7]. En referència als seus estàndards, n'hi ha de molts tipus diferents, emmarcats en diferents branques. La branca que conté l'estàndard CCSDS-123 és la de l'Àrea de Serveis de Connexió a l'Espai, que s'encarrega de crear sistemes que facilitin la comunicació entre les diferents entitats a l'espai. La primera publicació que es va fer en referència a la compressió lossless d'imatges de remote sensing va ser el maig del 1997[8], on es feia la primera menció a l'estàndard de compressió de dades. Després es van fer noves publicacions, ampliant i millorant l'estàndard esmentat, com a la primera versió del 1997del CCSDS-121.0, les corresponents correccions i la versió final actualitzada.[9,10,11,12].

Després d'establir l'estàndard per la compressió de dades, es va presentar l'estàndard de compressió *lossless* d'imatges al 2005[13], el qual s'ha anat corregint fins l'any 2011, un any abans que aparegués la primera recomanació de l'estàndard CCSDS-123 [5]. Aquest estàndard permet la compressió lossless d'imatges multicomponent, explotant

- *E-mail de contacte: alex.bonillo@e-campus.uab.cat*
- *Menció realitzada: Computació*
- *Treball tutoritzat per: Joan Bartrina (DEIC)*

la redundància entre les diferents bandes i la redundància interna de cada banda. Es pot veure com el codificador més complet per a lossless del CCSDS.

L'any 1979 Rissanen i Langdon trabllant per IBM van presentar el codificador aritmètic per primer cop[14]. En aquesta publicació s'estableix el que es la codificació aritmètica bàsica. Tot i això, a partir de la implementació bàsica, han anat sorgint optimitzacions que han permès millorar el rendiment del codificador aritmètic bàsic. Així, per exemple existeixen codificadors com el MQ coder utilitzat en aquest projecte[15] que aconsegueixen millors ratios de compressió. El paquet DumbMQ desenvolupat per GICI implementa una versió del Q coder.

4 METODOLOGIA I TASQUES

En aquest apartat s'esmentaran totes les tasques que s'han realitzat al llarg del projecte i que han permès complir 3 sub-objectius marcats des d'un principi.

En general, les tasques realitzades es divideixen segons els objectius que s'han esmentat a l'apartat anterior.

4.1 Iniciació QEmpordà

4.1.1 Iniciació del treball

Com ja s'ha comentat en els apartats anteriors, aquest era un treball que no es feia des de zero. El principal objectiu era afegir certes funcionalitats al QEmpordà per tal que millorés els seus resultats de compressió, de forma que s'havia de treballar sobre un projecte que ja s'havia construït.

Així doncs, una part important del projecte va ser, inicialment conèixer quin era el context en que les tasques es realitzaven. Van ser molt importants les reunions inicials, en que es va conèixer l'existència de la implementació de l'estàndard CCSDS-123, i es van conèixer les expectatives inicials del treball. En un principi els objectius es van marcar al voltant de la programació de més d'un mòdul codificador enllaçat al QEmpordà. Aquests mòduls eren, a més de la variant DumbMQ del codificador aritmètic[15], el codificador PACO[17] una variant del Q coder amb paraules de mida fixa i el codificador aritmètic basat en contextos de l'estàndard de codificació de vídeo H.264[16].

A més, com el projecte s'ha realitzat amb la estreta col·laboració del GICI, era important també conèixer a partir de les reunions inicials, quins eren els mitjans de què es disposava per tal de realitzar el treball. A més era important saber com, mitjançant aquests recursos disponibles es podia establir una metodologia que permetés arribar a complir els objectius finals desitjats.

4.1.2 Documentació

Tot i que sembla evident, una de les parts fonamentals de l'inici del projecte que van ajudar molt a l'hora de complir el primer objectiu del projecte de forma satisfactòria va ser l'obtenció i posterior estudi de la documentació referent, en primer lloc a l'aplicació QEmpordà i per últim a la codificació aritmètica.

Així doncs, a la primera reunió del projecte es va veure de forma genèrica quins eren els objectius que s'esperaven complir durant la realització d'aquest projecte. Això va comportar una visió prou general del que era l'aplicació QEmpordà i com funcionava i una primera visió de temes de codificació aritmètica. En aquesta primera reunió es van obtenir documents que més endavant resultarien molt importants per tal de comprendre correctament el funcionament dels codificadors aritmètics[18]. Aquests documents eren majoritàriament exercicis utilitzats per els professors del departament per donar les explicacions de codificació aritmètica als alumnes de grau i enunciats de pràctiques on es podien observar exemples molt clarificadors de què és i com funciona un codificador aritmètic.

Gràcies a tots aquests articles i documents es va adquirir una millor comprensió del funcionament de l'aplicació QEmpordà amb la qual s'havia de treballar i de la codificació aritmètica que era el fonament del projecte.

4.1.3 Presa de contacte amb el codi font

Un cop es va fer l'estudi de la documentació corresponent i es va obtenir un coneixement fonamentat sobre el que era la codificació aritmètica i es va llegir sobre el funcionament i l'estructura interna de l'aplicació QEmpordà, ja es podia començar a programar, per tal de veure de quina forma es podrien realitzar els objectius següents. Abans d'entrar en detalls dels canvis realitzats en l'aplicació cal proveir el lector d'una visió breu de l'estructura i el funcionament de l'aplicació.

Estructura

Com ja hem dit, el QEmpordà és una implementació de l'estàndard CCSDS-123. Aquesta funciona amb la aplicació *emporda.jar* que es troba dintre del directori *dist* a la carpeta de l'aplicació[6]. De totes formes es pot accedir també al codi font, per tal de fer compilacions i execucions amb la comanda *ant* de Java.

D'altra banda, pel que fa a l'estructura, podem observar com el QEmpordà té dos components principals: el compressor i el descompressor. La figura 1 mostra l'esquema que representa l'estructura interna del QEmpordà, la qual com es pot apreciar consta de diferents mòduls que permeten realitzar diferents operacions sobre les imatges. Aquesta implementació modular de l'aplicació ha estat fonamental per poder realitzar la implementació del nou mòdul codificador amb facilitat.

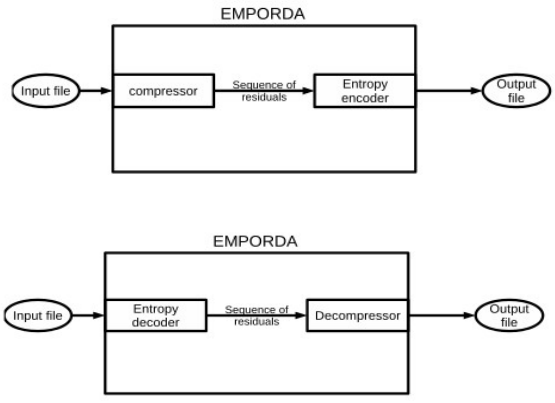

Figura 1: Esquemes abstractes de l'aplicació QEmpordà del procés de compressió i descompressió. Disponibles a: http://gici.uab.cat/GiciApps/EmpordaManual.pdf

Execució

A l'hora de fer l'execució s'ha de tenir en compte els requeriments hardware i software. En referència al Hardware s'ha de tenir en compte la quantitat memòria del sistema disponible abans de fer l'execució. Així la quantitat de memòria necessària depèn de la imatge que es vulgui comprimir, cosa que pot resultar un problema, tenint en compte que les imatges que es processen podrien arribar a ser molt grans. Sobre els requeriments de software només cal tenir en compte que la versió de les llibreries de Java usades per fer la implementació del QEmpordà és la JSE v. 1.6, de forma que es pot executar amb versions del JRE iguals o superiors a la versió 1.6.

Un cop coneguts els requeriments només cal conèixer el format de l'execució, la qual es pot fer des d'una terminal. A les següents taules es poden observar les sintaxis de les crides a la part de compressió i descompressió del QEmpordà.

```
$java -jar emporda.jar -i inputImageName -ig
numberOfBands numberOfRows numberOfColumns 3
0 0 -o outputFileName -c -ec Encoder Type [-pt AC
option] -f optionFileName -v
```

```
$java -jar emporda.jar -i inputCompressedFile -o
outputImageDecompressed -d -v
```

Després de conèixer tots aquests detalls es va procedir a realitzar les primeres proves amb el codi font. Aquestes proves consistien principalment en intentar afegir paràmetres nous i aconseguir que el fet d'afegir-los no espatllés les capçaleres de les imatges i es pogués seguir realitzant la compressió i descompressió correctament.

En cas de tenir algun dubte amb l'explicat a aquest apartat, es pot consultar informació més ampliada al manual del QEmpordà[19]

4.2 Implementació del codificador aritmètic

Un cop s'havia finalitzat el procés de documentació i les bases teòriques sobre la codificació aritmètica, els coneixements eren ja suficients com per afrontar la realització d'aquest projecte. En aquest punt es va procedir a començar la part de programació pròpiament dita. Aquesta gran tasca consta de dues subtasques que són la implementació de l'enllaç entre el nou mòdul corresponent al codificador aritmètic i el QEmpordà i les primeres proves realitzades per tal de veure si el procés era *lossless*.

4.2.1 Implementació codificador aritmètic

Abans d'entrar en detalls de la forma en que es van enllaçar el nou mòdul del codificador aritmètic i el QEmpordà, cal explicar breument en que consisteix un codificador aritmètic genèric.

Fonaments de la codificació aritmètica

Com molt bé ve definit al següent report de pràctiques curriculars del DEIC: "*A diferència d'altres mètodes de compressió, com ara el mètode Huffman[20], la codificació aritmètica consisteix en assignar a cada missatges, i no a cada símbol, un valor codificat, entenent per símbol un element del nostre alfabet A i per missatge un conjunt de símbols. Aquest valor codificat consisteix en un número de la recta real que es troba a l'interval [0,1).*".

Així doncs, si tenim en compte un alfabet $A = \{a_1, \cdots, a_n\}$ i la seva distribució de probabilitats P corresponent es poden crear taules de probabilitat com la de la següent figura:

Símbol	prob[i]	Prob[i]	Rang
a_1	p_1	$P_1 = p_1$	$[0, P_1)$
a_2	p_2	$P_2 = p_1 + p_2$	$[P_1, P_2)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
a_i	p_i	$P_i = p_1 + \cdots + p_i$	$[P_{i-1}, P_i)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
a_n	p_n	$P_n = p_1 + \cdots + p_n = 1$	$[P_{n-1}, P_n)$

Figura 2: taula de probabilitats exemple. Cortesia de: Fernando García i Joan Bartrina

Podem observar, com en aquest taula hi ha d'haver el valor de símbol i la seva probabilitat, a més de la probabilitat acumulada i un rang de probabilitats. És molt important saber que aquesta taula de probabilitats és el fonament de la codificació aritmètica i ha de ser igual tant en la part de codificació com de descodificació, de no ser així el procés mai podrà ser simètric.

D'altra banda, com senyalen Fernando García i Joan Batrina al report esmentat anteriorment "*L'interval [0,1) inicial es divideix en n subintervals corresponents als* n *símbols de l'alfabet i proporcionals a la mida de les seves probabilitats:* $[0,P_1),[P_1,P_2),\cdots,[P_{n-1},1)$." Aleshores segons el símbol que

s'hagi de codificar, s'ha d'anar a la part de l'interval corresponent per anar construint el valor real que finalment equivaldrà al missatge codificat.

Així doncs, per cadascun dels símbols a_i que entren a codificar-se, s'haurà d'accedir a l'interval corresponent $[P_{i-1},P_i)$ i tornar a dividir aquest interval en n rangs com en el cas anterior, tot i que ara els valors extrems dels intervals seran els corresponents al rang accedit anteriorment. D'aquesta manera a mesura que vagin entrant els símbols del missatge, s'aniran creant i accedint nous intervals que finalment, un cop s'han codificat tots els símbols esdevindrà en un interval de probabilitats on qualsevol component representarà la codificació del missatge. A la figura 3 es pot observar un exemple del procés de codificació aritmètica.

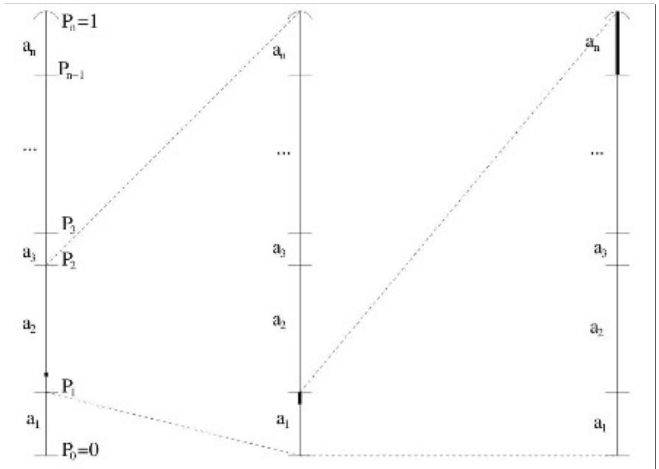

Figura 3: Exemple de procés de codificació aritmètica. Imatge obtinguda a report pràctica curricular DEIC[21]

Implementació

A l'hora de fer la implementació s'havien de tenir en compte les característiques tant de l'aplicació QEmpordà, com del mòdul codificador que s'havia d'utilitzar, en aquest cas el mòdul DumbMQ. La implementació d'aquest mòdul es podria resumir en 3 parts fonamentals:

1) Preparació QEmpordà

En primer lloc, el que s'havia de fer era preparar l'aplicació QEmpordà per tal de rebre una nova opció de codificació. Com s'ha comentat anteriorment, l'estàndard CCSDS-123 recomana dues opcions de codificació que són l'*Entropy Coder[22]* i el *Block Entropy Coder[23]*, de forma que es podria pensar que l'objectiu seria afegir una nova opció a un sistema que, per defecte deixa escollir entre dues opcions. Això però es va haver d'implementar, és a dir, primer permetre escollir enter les dues primeres opcions i després afegir-ne una altra per poder escollir el nou codificador aritmètic. A més, com veurem a les següents seccions es van implementar diverses formes diferents per fer funcionar el codificador aritmètic, de forma que es va haver d'implementar un paràmetres extra per tal d'escollir quina opció era desitjada (observar paràmtres *ec* i *pt* a la figura a la sintaxi mostrada a l'apartat execució).

2) Adaptació mòdul Q coder

A continuació el que s'havia de fer era preparar el nou mòdul codificador que havia estat donat, per tal que funcionés amb l'aplicació QEmpordà sense donar problemes. Per a fer això, s'havia de tenir en compte una característica molt important de l'aplicació QEmpordà que és que, per tal que es respecti l'estructura entre classes desitjada, existeixen diverses interfícies o "classes esquelet" que els diferents mòduls havien d'implementar o estendre. Així doncs, es van haver d'implementar alguns mètodes dins del mòdul codificador AC, per tal que entrés en correspondència amb l'estructura de classes esmentada, cosa que no va resultar gaire complexa.

3) Enllaçament

Per acabar només s'havien de construir els nous codificador i descodificador aritmètics dins de l'aplicació general i fer la crida als mètodes codificadors o descodificadors. Aquesta però va ser la tasca més complexa, ja que hi va haver un element que no es va tenir en compte fins pràcticament el final: les taules de probabilitats. El problema doncs, era implementar les taules de probabilitat per tal que el codificador aritmètic funcionés correctament. Per construir les taules de probabilitat de forma correcte, es va haver d'esbrinar quin era el format d'aquestes esperat pel mòdul DumbMQ. Fins que no es va descobrir que les taules de probabilitat estaven construïdes sobre llistes de probabilitats acumulades i que s'havien d'utilitzar mètodes que venien amb el propi mòdul codificador nou, es van fer moltes proves i es van requerir moltes hores.

Un cop realitzats tots aquests passos ja es podien fer les primeres proves amb el nou mòdul i observar el seu rendiment.

4.2.2 Primeres proves amb l'AC

Les primeres proves que es van realitzar amb el mòdul de codificació aritmètica no es van fer amb l'objectiu d'obtenir uns grans resultats, ja que s'entenia que el procés de creació de les taules de probabilitat era quelcom complex i que s'hauria de fer més recerca abans de trobar uns resultats acceptables. L'objectiu principal doncs, era veure que el procés fos sense pèrdua, és a dir, que la imatge original i la descodificada fossin iguals.

A la figura 4 es pot observar mitjançant la comanda *cmp* d'un terminal Linux, com el procés és realment *lossless* i en segon lloc, es poden veure els resultats que es van obtenir pel que fa a eficiència fent una comparació amb els codificadors per defecte.

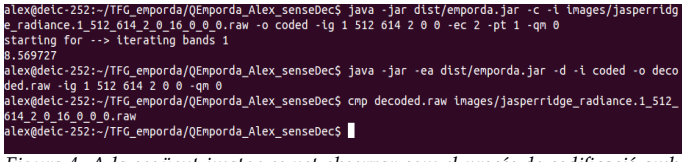

Figura 4: A la següent imatge es pot observar com el procés de codificació amb el nou mòdul AC és lossless

	1 banda	224 bandes
Entropy Integer	7,885132	5,997016
Block Adaptive	8,118042	6,122948
AC – no optimitzat	16,080572	16,010864

Taula 1: Eficiència dels 3 codificadors implementats

4.3 Millores proposades i anàlisi de resultats

Com s'ha comentat a la secció anterior era d'esperar que sent la creació de taules de probabilitat una tasca complexa, els resultats obtinguts en les primeres proves no milloressin els resultats dels codificadors per defecte del CCSDS-123. Així doncs, després d'aconseguir que el procés de compressió i descompressió fos *lossless* el que tocava fer era cercar optimitzacions del nou mòdul codificador per tal de millorar aquests resultats obtinguts.

4.6 Millores Proposades

Les millores que es proposen en aquest apartat es centren únicament en millorar el rendiment de compressió del CCSDS-123 (obtenir fitxers més petits), en aquest treball els temps de càlcul és irrellevant.

Les millores proposades van girar totes entorn a la creació i la gestió de les taules de probabilitat. Hi ha hagut diverses alternatives que s'han provat però en aquest apartat només se n'esmenten les més interessants pel que fa a resultats obtinguts.

Per tal d'entendre les optimitzacions i modificacions que es van fer s'ha d'entendre quin va ser el punt de partida, és a dir, quina era la forma de les taules de probabilitat quan es va iniciar aquest procés. Aquesta opció es va programar al principi per fer proves de funcionament únicament i consistia en crear una taula de probabilitats equiprobable amb un valor de creació molt gran (2^1) i sense tenir en compte absolutament per a res els símbols que apareixien a la imatge. Crear taules de probabilitat equiprobables significa que, per tal de crear les probabilitat s'assigna la mateixa freqüència (nombre d'aparicions a la imatge) a tots i cadascun dels símbols que es tenen en compte, de forma que aquests acaben tenint la mateixa probabilitat inicial d'aparèixer. A partir d'aquest punt es van proposar una sèrie de variants.

Taules de probabilitat amb valors predits ordenats (OPSI): La primera opció consistia amb una taula de probabilitat amb el mateix format que la opció inicial, és a dir, la llargària de la taula coincidia amb el valor del símbol màxim (valor d'intensitat) que apareixia a la imatge. Aquesta opció però tenia dues diferències fonamentals amb la opció inicial. En primer lloc, la llargària de la taula era molt més reduïda, ja que en aquest cas es tenien en compte els valors apareguts a la imatge (valors originals de la imatge passats per la part de predicció o *predictor*), no s'utilitzava un número d'un valor molt gran sense cap sentit. En segon lloc, les taules que es creaven, d'una banda no assignaven probabilitat a aquells símbols que no apareixien dintre dels valors predits, és a dir, no es tenien en compte valors no apareguts; i d'altra banda, no es creaven taules de probabilitat equiprobables, ja que s'assignava la freqüència real a tots els valors predits i no només una aparició, com en el cas anterior.

Taules de probabilitats amb histogrames de banda (HSI): Es pot observar com, en el mètode anterior, al final el que s'està fent és emmagatzemar la imatge predita de forma ordenada, cosa que pot semblar força ineficient. Sí que és cert que hi ha opcions més eficients, com per exemple la que es presenta aquí.

En aquest cas el que es fa no és més que crear les taules de probabilitat a partir dels histogrames de la imatge, és a dir, en compte d'emmagatzemar els valors predits ordenats, s'emmagatzema cadascun dels símbols que apareix a la imatge amb el seu nombre d'aparicions al costat. És evident que aquest mètode és molt més eficient que l'anterior, només pel fet que s'emmagatzema molta menys informació necessària després per crear les taules de probabilitat. A més, d'aquesta manera crear les taules de freqüència amb que després es creen les taules de probabilitat és molt més senzill, ja que els histogrames no són més que freqüències.

Taules de probabilitats amb histogrames generals (GHSI): Durant la implementació de les taules de probabilitats esmentades anteriorment, es va tenir una idea que podia ser interessant a l'hora de millorar els resultats. Durant les primeres proves amb les taules anteriors es van observar els histogrames que resultaven en les bandes de les imatges i es va veure que molts d'ells tenien distribucions de freqüència semblants. Per tal de fer aquestes observacions es va implementar un petit script que permetia crear les gràfiques dels histogrames mitjançant GNU-Plot (veure apartat 1 annex). Així, per exemple, una gran part de les imatges tenien histogrames que seguien un patró en el qual els símbols més petits iniciant per 0 tenien una major quantitat d'aparicions que no pas els símbols més grans.

Això va fer pensar que potser, si es creava un model matemàtic que intentés representar una distribució de freqüències única, emmagatzemant una molt petita porció d'informació es podrien obtenir millors resultats d'eficiència.

Després de pensar-ho detingudament es va decidir que la millor forma de crear aquest model matemàtic que representés una distribució de probabilitats general era crear un histograma general. Un histograma general no és més que la unió de tots els histogrames de les bandes de les imatges a comprimir, units en un gran histograma. Així doncs, imaginem que tenim una imatge AVIRIS de 224 bandes[24], diríem doncs que l'histograma general

d'aquesta imatge és crear un histograma gegant a partir de tots els histogrames de les 224 bandes, de forma que e aquest histograma general hi apareixerien tots els símbols i totes les vegades que aquests han aparegut durant les 224 bandes d'aquesta imatge.

Així, per tal de crear aquests histogrames generals es va decidir implementar una petita aplicació externa anomenada *hist creator*. Aquesta aplicació el que permet és obtenir l'histograma general de tots els histogrames continguts dins d'un directori, i es fa de la següent manera:

En primer lloc ens hem de col·locar dins de la carpeta de l'aplicació i dins del directori *src*,. A continuació s'ha de compilar el fitxer principal *App_main.java*, mitjançant el compilador de Java des de terminal de Linux i executar amb la comanda *java* el fitxer compilat *App_main* amb 3 arguments. Aquests arguments es corresponen amb el *path* d'on es volen llegir els histogrames, el nom que volem pel fitxer de sortida o o*utput* i finalment un *string* que sigui 'b' o 't' per establir si volem l'histograma general en binari o en format text. A més, en cas de no saber com funciona la crida a l'aplicació podem passar el paràmetre '-h' que mostrarà per pantalla la informació d'ajuda necessària per tal de poder fer la crida correctament. A la figura 5 es pot veure un exemple d'una crida a l'aplicació. Per acabar si es volen observar els *outputs* només ens hem d'adreçar a la carpeta *outputs* dins del directori de l'aplicació.

```
alex@deic-252:~/TFG_emporda/hist_creator$ cd src
alex@deic-252:~/TFG_emporda/hist_creator/src$ javac App_main.java
alex@deic-252:~/TFG_emporda/hist_creator/src$ java App_main /home/alex/TFG_emporda/QEmporda_Alex_sense
Dec/histogrames/test_hists/ output_radiance t
```

Figura 5: Exemple de la crida al hist creator, l'aplicació per crear els histogrames generals

Un cop s'havia creat l'histograma general, només quedava implementar la creació de taules de probabilitats a partir d'aquest, cosa que ja estava pràcticament feta, degut al procés de creació anterior que també es realitzava a partir d'histogrames amb el mateix format.

Les següents propostes funcionen com a complements de les anteriors configuracions de les taules de probabilitat.

Imatges simplificades (HSI-LUT): Una característica de les imatges que es tenien per fer proves (AVIRIS[24] i IASI[25]) és que els valors resultants de les prediccions tenien una concentració d'aparicions molts grans en els símbols més petits i una concentració més petita en els símbols més grans. A més, donava la casualitat que els símbols més grans que apareixien tenien valors molt grans. Així potser, hi havia imatges en que el símbol 65000 apareixia una vegada i que el fet que la mida de les taules de probabilitat depengués directament del valor més gran aparegut, feia que es creessin taules innecessàriament llargues i amb la gran majoria de posicions buides (posicions corresponents a símbols que ni apareixien a la imatge). Aquest problema podia afectar en que a l'hora d'escriure els valors dels histogrames s'havien d'utilitzar més bits dels necessaris.

Per tal de millorar aquest problema el que es va fer va ser simplificar les imatges predites mitjançant una *look-up-table*, de forma que els símbols predits de la imatge automàticament passarien a ser la posició d'aquest símbol en una llista ordenada amb els símbols predits ordenats. A continuació, un exemple molt senzill de la simplificació d'una imatge:

Imaginem una imatge de mida 2x2 amb els següents valors d'intensitat

1	**2**
3	**65000**

Que simplificada passaria a ser:

1	**2**
3	**4**

D'aquesta forma veiem com cada símbol pren el seu valor dintre d'una llista ordenada de símbols de major a menor, d'aquí que el 65000 obtingui el valor 4, ja que seria el 4rt element d'una llista ordenada dels símbols. S'aconsegueix doncs, que els valors a codificar siguin molt més petits i en cap cas es repetiran per que s'assigna el seu ordre en una llista de símbols **diferents** ordenats de major a menor, com hem dit anteriorment. D'altra banda, a l'exemple anterior, per fer el procés invers només es necessari emmagatzemar una *look-up-table* que és una taula que farà tornar la imatge a l'estat inicial. A l'exemple anterior la *look-up-table* seria així:

1
2
3
65000

Amb aquesta taula si ens fixem, el valor d'intensitat de la imatge simplificada, representa **l'índex o posició a la taula** del seu valor original, de forma que només s'haurà de fer un simple accés a la taula *look-up* per tal d'adquirir el valor inicial d'un píxel concret.

Taules de probabilitats dinàmiques[26]: una altra prova va ser la de dinamitzar les taules de probabilitat. Aquesta actualització de les taules, que actua com a complement de les configuracions esmentades anteriorment, consisteix en anar actualitzant els valors de les taules de freqüència a mesura que van entrant nous símbols a codificar. Així doncs, en un principi es formen les taules de probabilitat, que en aquest cas pot ser a partir dels histogrames o els valors predits ordenats, però a més, després es pensava que s'afegiria un grau més de precisió a aquestes taules fent una actualització periòdica d'aquestes.

	Entropia	C 1	C 2	NSI	HSI	simplified	simp. With Lut	OPSI	GHSI
cuprite_radiance	5,33	6,00	6,12	5,50	5,56	5,54	5,63	5,57	6,35
cuprite_reflectance	6,78	7,44	7,50	7,25	7,35	7,24	7,40	7,36	8,12
hawaii_uncalibrated	2,63	2,72	2,93	2,70	2,71	2,71	2,71	2,71	3,08
jasperridge_reflectanc	8,03	9,07	8,81	8,83	9,12	8,43	9,21	9,10	9,57
lowaltitude_radiance	6,46	8,23	8,05	6,90	7,05	6,96	7,19	7,06	7,59
lunarlake_radiance	5,22	5,83	5,88	5,36	5,41	5,38	5,44	5,39	6,10
lunarlake_reflectance	6,09	6,33	6,47	6,41	6,48	6,38	6,49	6,49	7,49
Maine_uncalibrated	2,73	2,80	3,00	2,80	2,81	2,81	2,81	2,81	3,16
moffetfield_radiance	5,85	6,71	6,66	6,06	6,13	6,10	6,21	6,15	6,78
moffetfield_reflectanc	6,80	7,25	7,26	7,34	7,50	7,29	7,54	7,50	8,18
yellowstone_radiance	4,59	5,70	5,45	4,76	4,82	4,78	4,88	4,83	5,26
yellowstone_uncalibr	6,33	6,40	6,53	6,43	6,43	6,46	6,49	6,46	6,79
Mitjanes	**5,57**	**6,21**	**6,22**	**5,86**	**5,95**	**5,84**	**6,00**	**5,95**	**6,54**

Taula 2: Taula de resultats generals amb tots els mòduls codificadors provats

Per tal de fer la implementació el que es va fer és crear una finestra que marcava cada quants elements s'havia de fer el canvi a les freqüències dels símbols. Aquesta finestra, no era més que un vector on s'anaven emmagatzemant tots els símbols que entraven i quan aquest vector estava ple, s'enviava a la taula de probabilitats per tal que es fes una actualització de les freqüències amb els símbols que hi havia dins d'aquest.

Aquesta finestra tenia una mida regulable que va permetre fer un millor anàlisi del rendiment d'aquest tipus de taules.

4.7 Obtenció de resultats

Un cop implementades totes les millores plantejades inicialment, era hora de començar a provar quins eren els resultats que donaven.

Aquesta part del projecte va tenir moltes fases similars, tot i que s'estaven fent proves amb configuracions diferents del mòdul que implementa el codificador aritmètic dins del CCSDS-123. Al cap i a la fi, les proves consistien sempre en el mateix, és a dir, agafar totes les imatges a provar del directori on es trobaven contingudes i fer la crida al QEmpordà fent petits canvis a l'hora d'escollir el tipus de codificador. Hi havia ocasions en que potser s'havien de fer petites modificacions al codi, però en principi no tenen importància. En aquesta fase es van utilitzar diversos scripts amb llenguatge *bash*, que van ajudar molt a realitzar les tasques de forma automatitzada i fins i tot poder fer més d'una tasca mentre s'estaven fent execucions de prova. Alguns d'aquests scripts venen continguts al primer apartat de l'annex.

L'única part on hi va haver una mica més de canvi va ser a l'hora de fer proves amb les taules dinàmiques, les quals requerien canvis al codi, comentant algunes seccions del mateix i modificant paràmetres com la finestra, que no es poden modificar des de la línia de comandes.

5 RESULTATS

Les proves que s'han realitzat en finalitzar aquest projecte i que permeten analitzar el rendiment final de les implementacions i modificacions realitzades, han estat fetes sobre dos tipus d'imatges diferents.

Aquestes són les imatges obtingudes amb el satèl·lit AVIRIS[24] i les imatges obtingudes amb el satèl·lit meteorològic IASI[25].

Les proves més rellevants que s'han realitzat en aquest projecte es corresponen amb les millores i configuracions que s'han esmentat a la secció anterior, de forma que serà dels resultats d'aquestes configuracions del que es parlarà aquí. Tot i això aquestes queden dividides segons les imatges amb que s'han analitzat, com es veu a continuació:

Resultats amb imatges AVIRIS: La taula 2 mostra els resultats obtinguts per les imatges AVIRIS amb les diferents millores proposades OPSI, HSI, GHSI i HSI-LUT. Cal que quedi clar que les columnes C1 i C2 corresponen als codificadors implementats per defecte, és a dir, *l'Entropy Integer Coder* i *el Block Adaptive Coder* respectivament.

Mirant la mitjana dels rendiments, es pot observar que el millor rendiment s'obté amb la configuració HSI, que crea les taules de probabilitat a partir dels histogrames de les bandes. Aquest obté una millora aproximada d'una **4,1%** sobre el codificador per entropia i d'una **4,3%** sobre el codificador per blocs, quedant-se només 0,38 bpppb per sota de l'entropia, que representa el valor ideal. S'ha de dir, però que la configuració OPSI, corresponent als valors predits aconsegueix un resultats pràcticament iguals, millorant també d'una forma significativa els resultats obtinguts pels codificadors per defecte. No millora tant els resultats el procés de simplificació de la imatge combinat amb la configuració HSI, però encara podem observar com s'obté una millora del **3,4%** i **3,6%** sobre els codificadors per defecte.

Per acabar tenim la configuració GHSI que no obté uns bons resultats de rendiment, degut a un mal funcionament probablement del model de distribució de freqüències.

IMATGES	EC 0 – s16	EC 1 – s16	EC -HSI – s16	EC -HSI simp s.16	EC 0 – u16	EC 1 – u16	EC -HSI – u16	EC -HSI simp u.16
50Z	6,60	6,79	7,27	7,30	7,62	7,73	9,00	8,98
49Z	6,62	6,80	7,29	7,33	7,60	7,70	8,78	8,75
19Z	6,64	6,82	7,26	7,30	7,62	7,71	8,71	8,69
13Z	6,60	6,79	7,27	7,30	7,20	7,33	8,11	8,11
07Z	6,56	6,76	7,22	7,25	7,12	7,25	8,03	8,03
57Z	6,61	6,80	7,27	7,31	7,23	7,36	8,12	8,13
average	**6,60**	**6,79**	**7,26**	**7,30**	**7,40**	**7,51**	**8,46**	**8,45**

Taula 3: Resultats obtinguts amb les imatges IASI

Resultats amb imatges IASI: Aquests resultats es poden observar a la taula 3. En aquest cas no s'han fet proves amb totes les configuracions, ja que ja s'havia vist de proves anteriors que el millor rendiment el proporcionava la configuració HSI i, per tant només es volia mirar si també hi havia millora en aquest tipus d'imatges. A més, es va pensar que el rendiment de la simplificació d'imatges podia donar una millora més gran en aquest tipus d'imatges, degut a que s'havia vist que aquestes tenien uns símbols amb valors molt grans, i una molt gran quantitat de símbols intermitjos que no apareixien. Es van fer proves tenint en compte que les imatges eren de 16 bits amb signe i sense, per veure els efectes d'aquest canvi, però tant en el cas d'HSI, com en el cas de la simplificació d'imatges els resultats no milloren el rendiment dels codificadors que ja venien implementats per defecte.

Resultats amb taules dinàmiques: Les últimes proves que es van realitzar corresponen a les taules de probabilitats dinàmiques. Es van fer proves amb diversos valors de finestra com es pot veure a la taula 4, les quals s'executaven sobre **una banda d'una de les imatges del conjunt AVIRIS (A2 Annex)**, en concret l'imatge *cuprite radiance*. Per tal de tenir la referència del rendiment del codificador aritmètic amb millor rendiment en aquella imatge en concret (HSI) s'ha establert el resultat de la prova amb 512 com a valor de la finestra, que representa una mida de 512x512. D'aquesta manera un cop la finestra s'ha omplert amb tots els símbols per actualitzar, la codificació ha acabat i ja no té sentit actualitzar la taula de probabilitats.

A grans trets el que es pot observar són dues coses: En primer lloc és que, a mesura que la finestra es fa més petita els resultats empitjoren i, no només això, sinó que, en segon lloc, el temps requerit per fer la codificació de la banda augmenta significativament.

Window size	Time (s)	*bpppb*
512	0,62	7,87
256	320,97	8,95
128	418,00	14,82
64	437,00	21,27
32	437,24	22,80
16	443,27	22,46

Taula 4: Resultats amb taules de probabilitats dinàmiques

6 CONCLUSIONS

Amb la realització d'aquest projecte s'ha pogut treballar amb un mòdul de codificació aritmètica i, a més analitzar profundament el funcionament i les operacions que pateixen les imatges durant totes les part de la implementació donada de l'estàndard CCSDS-123. La majoria de conclusions que s'extreuen de la realització d'aquest projecte giren al voltant del rendiment d'aquest nou codificador amb diferents tipus d'imatges i amb diferents configuracions. Les més interessants són les següents:

- Pel que fa a les imatges obtingudes amb el satèl·lit AVIRIS, s'ha vist com s'han aconseguit millores significatives. Així doncs, com s'ha vist a la secció anterior hi ha hagut fins a 3 configuracions amb que s'ha aconseguit millorar l'eficiència de l'aplicació QEmpordà. En primer lloc tenim la configuració *HSI*, que és la que ha donat uns millors resultats d'eficiència i a l'hora d'executar és la més ràpida de les configuracions del nou mòdul AC. Després tenim l'*OPSI*, que també ens ha permès obtenir uns bons resultats de millora sobre els codificadors per defecte, però el seu gran inconvenient és la lentitud a l'hora de codificar les imatges. Per últim la simplificació de les imatges també ha resultat un mètode que ha millorat resultats.

- Hi ha tres tipus d'imatges AVIRIS, com es pot veure en el nom de les imatges a l'apartat de proves generals (taula 2). Aquestes són les imatges *radiance (figura 6)*, *reflectance (figura 7) i uncalibrated (figura 8)*, les quals es classifiquen segons el tractament que reben les imatges després de ser preses als satèl·lits. Un exemple d'aquestes imatges es pot observar a l'últim apartat de l'annex. Si s'observa la taula 2 un altre cop es podrà observar on els bons resultats dels codificadors anteriors s'obtenen en les imatges de radiància o *radiance*, de forma que el tractament que millor funciona amb les configuracions implementades és aquest. Val a dir, que no es pot saber exactament quin tipus de tractament reben aquestes imatges, ja que són operacions que realitza la NASA als seus satèl·lits i no se'n dóna informació.

- On no es troba millora és amb les imatges IASI, com es pot observar a la taula esmentada anteriorment. Aquestes imatges es caracteritzen per de no tenir tans espais 'buits' als valors d'intensitat, és a dir, el mín i el màx. d'intensitat estan més propers, a diferència de les imatges AVIRIS. Això el que provoca, segurament és que els codificadors per defecte tinguin un un molt millor rendiment i el nou mòdul implementat quedi en tercer lloc.

- Per últim només afegir que configuracions com les taules dinàmiques i els histogrames generals no han tingut un rendiment satisfactori com s'esperava. En primer lloc, pel que fa a les taules dinàmiques, és molt probable que no s'hagi fet una implementació correcta del procés, el que ha afectat directament a que no funcioni de forma òptima. D'altra banda, pel que fa als histogrames generals, es pot concloure que no es pot extreure una distribució de freqüències general que funcioni correctament amb aquest conjunt d'imatges, el que significa que no són tan similars entre elles com per unir-les en un únic model de freqüències.

Com a possibles millores o treballs futurs a realitzar en aquest projecte es podria dir:

- Millora en la implementació de la forma de dinamitzar les taules de freqüències. Sembla molt estrany com, una configuració que acostuma a donar bons resultats com és la dinamització de taules de probabilitat, en aquest cas empitjori tant els resultats.

- Optimització de les diferents configuracions del mòdul codificador AC, especialment la part de valors predits ordenats, la qual requereix d'un temps d'execució massa alt en comparació als altres mòduls.

- Proves amb altres mòduls codificadors, que podrien ser perfectament les versions alternatives al DumbMQ comentades a l'estat de l'art (PACO i H.264), amb les quals es podrien millorar encara més els resultats.

AGRAÏMENTS

Vull agrair, en primer lloc al Joan Bartrina, tutor d'aquest TFG, la seva ajuda durant tota l'execució d'aquest projecte, en forma d'idees, consells i documentació donada. A més vull mostrar el meu agraïment al *DEIC* i més concretament al grup *GICI*, per donar-me l'oportunitat de realitzar aquest projecte en un entorn de recerca idoni per haver tingut èxit en la realització del mateix.

BIBLIOGRAFIA

[1] Conferències On-Board Payload Data Compression Workshop Barcelona, Octubre del 2012. Disponible: http://www.congrexprojects.com/2012-events/12c17/introduction

[2] I. Blanes, E. Magli, J. Serra-Sagristà, "A tutorial on Image Compression for Optical Space Imaging Systems.". *IEEE geoscience and remote sensing magazine*, vol.2, Issue.3, p.8-26. Setembre del 2014.

[3] "The use of reprogramable FPGAs in space", ESA's website: http://www.esa.int/Our_Activities/Space_Engineering_Technology/Microelectronics/The_use_of_reprogrammable_FPGAs_in_space.

[4] Consultative Committee for Space Data Systems (CCSDS), *Lossless Data Compression* CCSDS 121.0-B-2, Blue Book. CCSDS, May 2012. Online]. Available: http://public.ccsds.org/publications/archive/121x0b2.pdf

[5] Consultative Committee for Space Data Systems (CCSDS), *Lossless Multispectral & Hyperspectral Image Compression* CCSDS 123.0- B-1, ser. Blue Book. CCSDS, May 2012. [Online]. Available: http://public.ccsds.org/publications/archive/123x0b1ec1.pdf

[6] GICI Group, Gici Emporda manual, Department of Information and Communications Engineering, UAB, Octubre 2011. Disponible: http://gici.uab.es/GiciWebPage/downloads.php#emporda

[7] About us page. CCSDS website. Extret el Juny del 2015 http://public.ccsds.org/about/default.aspx.

[8] Consultative Committee for Space Data Systems, *Image Data Compression* CCSDS 120.1-G-1, ser. Green Book. CCSDS, Maig. 1997

[9] Consultative Committee for Space Data Systems, Image Data Compression CCSDS 121.0-B-1, ser. Blue Book. CCSDS, Maig. 1997

[10] Consultative Committee for Space Data Systems, Image Data Compression CCSDS 121.0-B-1, ser. Blue Book. CCSDS, Maig. 1997, correcció tècnica 1. Nov.2006

[11] Consultative Committee for Space Data Systems, Image Data Compression CCSDS 121.0-B-1, ser. Blue Book. CCSDS, Maig. 1997, correcció tècnica 2. Set.2007

[12] Consultative Committee for Space Data Systems (CCSDS), *Lossless Data Compression* CCSDS 121.0-B-2, ser. Blue Book. CCSDS, May 2012. [Online]. Available: http://public.ccsds.org/publications/archive/121x0b2.pdf

[13] Consultative Committee for Space Data Systems (CCSDS), *Lossless Data Compression* CCSDS 122.0-B-1, ser. Blue Book. CCSDS, Nov.2005. [Online]. Available: http://public.ccsds.org/publications/archive/122x0b1c3.pdf

[14] Rissanen J.J. , Langdon, G.G. Arithmetic Coding, IBM J.R*es. Desv.23, 2(Març 1979), p. 149-162.*

[15] M. Slattery and J. Mitchell, "The Qx-coder", *IBM Journal of Research and development*, vol.42, no. 6, pp. 767-784, Nov. 1998.

[16] D.Marpe, H.Schwarz i T.Wiegand (Maig, 2003). Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard. IEEE, Berlin, Alemanya. Disponible: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1218195

[17] J.Tehuola I T.Raita (Maig, 1993). Arithmetic Coding into Fixed Length Codewords. IEEE, Turku, Finlàndia. Disponible: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=272486&tag=1

[18] P.G.Howard i J.S.Vitter, (Gener 1994). Arithmetic Coding for data Compression, IEEE, disponible: http://www.ittc.ku.edu/~jsv/Papers/HoV94.arithmetic_codingOfficial.pdf

[19] Manual QEmpordà, versió online: http://gici.uab.cat/GiciApps/EmpordaManual.pdf

[20] D. A. Huffman, "A method for the construction og minimum redundancy codes" *Proc.IRE* vol. 40, pp. 1098-1101, Setembre 1952

[21] F. García I J.Bartrina. Guió complert pràctica curricular de codificació aritmètica. Disponible: http://www.deic.uab.es/material/20351-guioComplet.pdf

[22] Lossless Data Compression, cCSDS-120.0-G-3 Green Book, Issue 3, Abril 2012. Disponible: http://public.ccsds.org/publications/archive/120x0g3.pdf

[23] Lossless Data Compression, cCSDS-121.0-B-2 Blue Book, Issue 2, Maig 2012. Disponible: http://public.ccsds.org/publications/archive/121x0b2.pdf

[24] satèl·lit AVIRIS lloc web. Versió consultada el 12 de Juny del 2015: http://aviris.jpl.nasa.gov/

[25] satèl·lit IASI lloc web. Versió consultada el 12 de Juny del 2015: http://www.eumetsat.int/website/home/Satellites/CurrentSatellites/Metop/MetopDesign/IASI/index.html

[26] A. Masmoudi. W.Puech. M.S. Bouhlel. Efficient adaptive arithmetic coding based on updated probability distribution for lossless image compression. Journal od Electronic Imaging (Digital). Volum 19(2) (Apr-Jun 2010). p:1-2. Disponible: http://www.lirmm.fr/~wpuech/recherche/publications/RI/10_JEI023014. Pdf

APÈNDIX

A1. SCRIPTS

a) Script utilitzat per obtenir informació de les imatge. Modificant el directori de les imatges es podia obtenir informació d'eficiència d'imatges AVIRIS i IASI.

```
rm data_test_AC.txt
clear
touch data_test_AC.txt
chmod 777 data_test_AC.txt



for image in `ls /home/alex/TFG_emporda/images/`; do
	echo "image: "$image
	echo "image: "$image >> data_test_AC.txt


	zsize=`echo $image | cut -d "." -f 2 | cut -d "_" -f 1`
	ysize=`echo $image | cut -d "." -f 2 | cut -d "_" -f 2`
	xsize=`echo $image | cut -d "." -f 2 | cut -d "_" -f 3`
	rm hist_*


	java -jar dist/emporda.jar -c -i <<path>> -o coded -ig $zsize $ysize $xsize 2 0 0 -ec 2 -pt 1 -qm 0 >> data_test_AC.txt


	bytesimage=`ls -all coded | cut -d " " -f 5`
	byteshist=0


	for j in `ls hist_*`; do
		lzma -e $j
		let "byteshist=byteshist+`ls -all $j.lzma | cut -d " " -f 5`"
		echo $byteshist
	done
	let "bytestotals_hist=bytesimage+byteshist"


	b_rate_hist=$(awk" BEGIN {printf \"%.9f\", $bytestotals_hist*8/$zsize/$ysize/$xsize}")


	echo "amb hists: "$b_rate_hist >> data_test_AC.txt
done
```

b) Script utilitzat per obtenir la representació gràfica d'alguns dels histogrames

```
num_files=`ls *.txt | wc -l`
echo "num_files: $num_files"
count=1 # grouping elements
lasts=4
grup=0
differ=0 # to calculate the tail elements
while [ "$grup" -lt "$num_files" ]
do
	let grup=$count\*4
	echo "grup: $grup"
	files=`ls *.txt | head -n $grup | tail -n $lasts`
	f_array=( $files )
gnuplot << EOF
	set style line 1 lc rgb '#0060ad' lt 1 lw 1 pt 7 ps 0.75   # --- blue
	set term postscript eps
	set output "result$count.eps"
	set size 1.0, 1.0
	set origin 0.0, 0.0
	set multiplot
	set size 0.5,0.5
	set origin 0.0,0.5
	plot "${f_array[0]}" title "title1" with linespoints ls 1
	set size 0.5,0.5
	set origin 0.0,0.0
	#set yrange [0:300]
	plot "${f_array[1]}" title "title2" with linespoints ls 1
	set size 0.5,0.5
	set origin 0.5,0.5
	plot "${f_array[2]}" title "title3" with linespoints ls 1
	set origin .5,0.
	plot "${f_array[3]}" title "title4" with linespoints ls 1
	unset multiplot
EOF
	let count=$count+1
	if [ "$grup" -gt "$num_files" ]
	then
		let differ=$grup-$num_files
	fi
	let lasts=4-$differ
done
```

A2. IMATGES AVIRIS

Radiance

Figura 6: Cuprite radiance

Reflectance

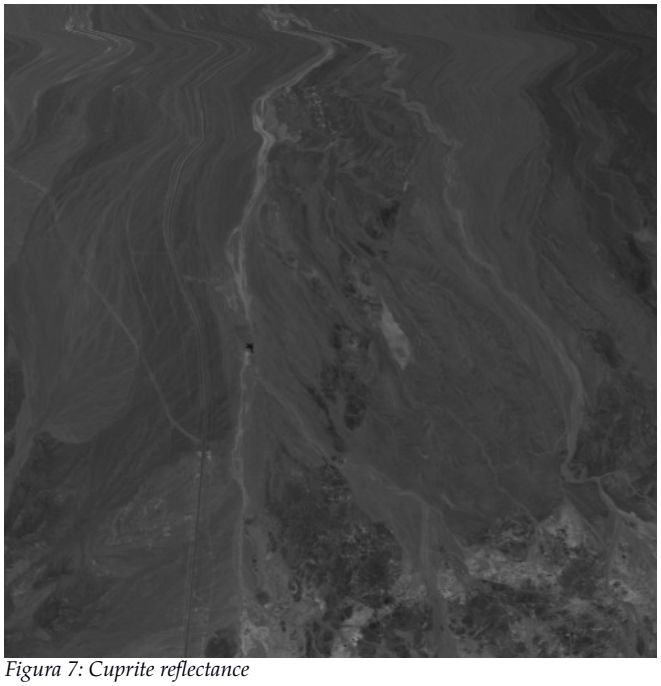

Figura 7: Cuprite reflectance

Uncalibrated

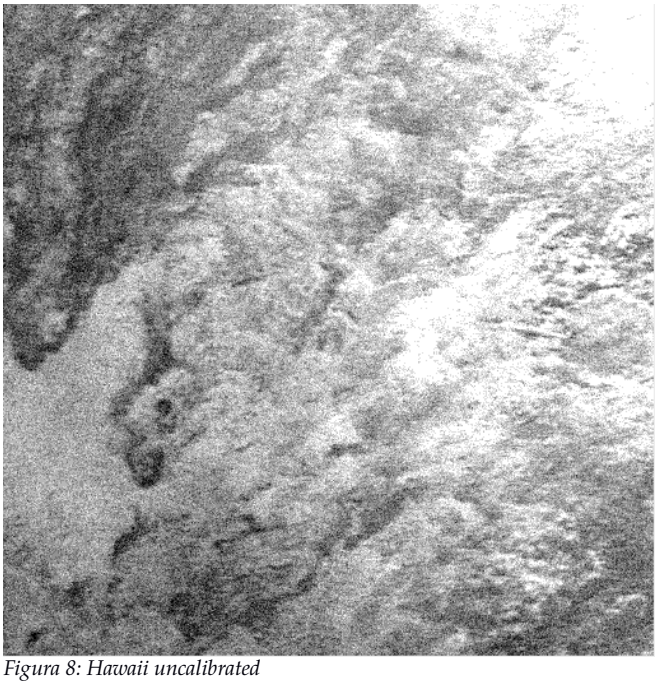

Figura 8: Hawaii uncalibrated