**Hierarchical Prediction and Context Adaptive**

**Coding for Lossless Color Image Compression**

**Abstract:** La compresion sin perdidas en RGB se consigue decorrelacionando con una transformada de color reversible. La componente Y se codifica con un metodo de compresión sin perdidas en escalas de grises convencional. Para la crominancia se ha desarrollado un esquema jerarquico que utiliza los pixels superior, izquierdo e inferior para la predicción cuando los metodos convencionales solo utilizan superior e izquierdo. Tambien se define un modelo de contexto para la prediccion de la señal de error que corresponde a cada contexto. Se reduce la compresión respecto a JPEG2000.

**Intro**

Debido a los requisitos de ancho de banda las imagenes se suelen comprimir en formatos con perdidas que tienen una gran compresión a costa de degradar la imagen. Hay casos en los que se quiere evitar la perdida de calidad de la imagen como son medicina, arte, prensa... Las camaras y las pantallas mejoran su calidad al bajar el precio de la memoria por eso queremos mantener la calidad de nuestras fotos artisticas librandonos de los artefactos de la compresion. Es una razon por la que la compresion sin perdidas se hace mas importante aunque las imagenes comprimidas con perdidas siguen valiendo para ciertos casos.

Algunos de los sistemas utilizados son (en modo sin perdidas) JPEG, JPEG-LS, LOCO-I, CALIC, JPEG2000, JPEG XR. Los metodos LOCO-I y CALIC se desarrollaron durante el proceso de estandizacion de JPEG por lo que muchas de las ideas del LOCO-I se aplicaron al formato JPEG-LS aunque CALIC proporciona una compresión mayor a costa de mas computaciones.

Para la compresion de imagenes en color lo componentes de color se decorrelacionan con una transformada de color y cada uno de los componentes transformados se comprimen independientemente con los metodos anteriores. Por ejemplo la transformada RGB a YCbCr se utiliza en compresion con perdidas para imagen y video. En el caso de la compresion sin perdidas hay muchas transformaciones que no se pueden aplicar porque no se pueden invertir aritmeticamente. Por lo tanto se definio una transformada de color reversible (RCT) para JPEG2000. Se ha investigado mucho para encontrar mejores RCTs y llegar a una que se aproxime mejor a la transformación YcbCr.

SE quiere desarrollar un esquema de prediccion jerarquica mientras que muchos de los metodos de prediccion para imagen sin perdidas estan basados en la prediccion del escaneo raster que muchas veces no es eficiente en regiones de alta frecuencia.

La predccion jerarquica ya fue propuesta pero solo se interpola un pixel. Aqui se diseña un predictor dirigido a bordes y un modelo adaptativo al contexto para este esquema jerarquico. Se propone un metodo que pueda usar los pixels de la fila inferior y los de arriba y la izquierda para la predicción del pixel codificado.

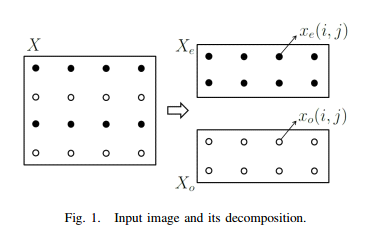
Para la compresion de imagenes en color el RGB se transforma primero en YCuCv por la RCT mencionada y el canal Y se codifica con un sistema de compresion convencional. En el caso de los canales de crominancia Cu y Cv la variacion de señal suele ser menor que en RGB pero sigue siendo alta cerca de los bordes. Para una predicción mas ajustada y para un modelo de predicción de errores mas exacto se utiliza el esquema jerarquico: La imagen de crominancia se descompone en 2 subimagenes. Ej: Las filas pares e impares. Cuando se codifica la fila par Xe podemos usar sus pixels para predecir el pixel de la fila impar Xo. Ademas las propiedades estadisticas de las dos subimagenes no son muy diferentes, el predictor de errores de una subimagen se puede modelar a partir del de la otra que contribuye a mejorar el contexto de modelado de la codificacion aritmetica.

Se ha experimentado con imagenes de varios tipos y se ha visto que el metodo propuesto da una compresion superior a JPEG2000 y JPEG-XR en muchos casos.

**Descomposicion jerarquica y prediccion de pixels**

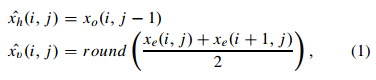
Los canales de crominancia Cu y Cv resultado de la RCT tienen diferentes estadisticas de Y y tambien de los planos de color RGB. En los canales de crominancia la variacion de señal total es suprime por la transformada de color pero la variacion sigue siendo alta en los bordes de los objetos. Los errores de prediccion en el canal de crominancia se reducen mucho en las regiones suaves pero siguen siendo altas en los bordes o en regiones con textura.

Para una compresion sin perdidas eficiente es importante estimar el error de prediccion para un modelo de contexto mejor incluso con una prediccion exacta. Por eso se propone un esquema de descomposicion jerarquica como este:



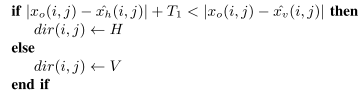
Los pixels de una imagen de entrada X se separan en dos subimagenes una imagen par Xe y una impar Xo. Primero se codifica Xe y se utiliza para predecir los pixels de Xo ademas Xe se utiliza para estimar las estadisticas de prediccion de errores de Xo. En la implementacion actual Xe se descompone una vez mas como se explica mas adelante.

Para la compresion de Xo utilizando Xe se emplea prediccion direccional para evitar grandes errores de prediccion cerca de los bordes. Para cada pixel Xo(i,j) el predictor horizontal xh(i,j) y el vertical xv(i,j) se definen como:



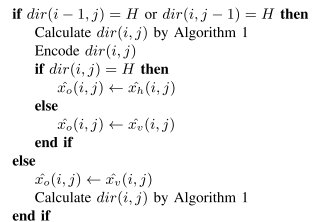
Uno de estos se selecciona como predictor de Xo(i,j). Con estos dos posibles predictores la aproximacion mas comun es “mode selection” donde se selecciona el mejor predictor para cada pixel y el modo (horizontal o vertical) se transmite como informacion complementaria. El predictor vertical suele ser mas correcto que el horizontal porque los pixels superior e inferior se utlizan para vertical mientra que para el horizontal solo esta el izquierdo. El horizontal solo es mas eficiente si los bordes son muy horizontales. Por ejemplo la frecuencia en la que se utiliza el predicto horizontal esta entre un 0.03% y un 1.45% de las imagenes probadas. El predictor vertical es el mas utilizado para muchos de los pixels y el modo de selección solo se utiliza cuando va a haber muchos bordes horizontales.

Para implementar esta idea se define una variable para la direccion del borde en cada pixel dir(i,j) que puede ser H o V. Actualmente solo sera H cuando el borde horizontal es muy fuerte y se aplicara V para el resto. Decidir esta direccion depende del algoritmo:



Se puede observar que la condicion para que el resultado sea H es menor que la condicion para que sea V.

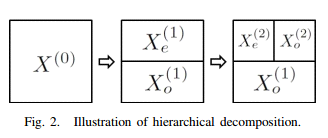
Basandonos en la direccion de los pixels el algoritmo de prediccion es:



Se puede ver que tambien es mas facil que se elija el modo V.

**Esquema de codificacion propuesto**

La imagen se transforma de RGB a YCuCv por una RCT. La imagen de luminancia Y se codifica con cualquier codificador sin perdidas como CALIC, JPEG-LS o JPEG2000. Las imagenes de crominancia Cu y Cv se codifican según los metodos que se acaban de ver. Para ser mas especifico la se descomponen en imagen pares e impares.



La subimagen X01 se predice y codifica mediante Xe1. La subimagen Xe1 se puede descomponer por columnas en una subimagen par Xe2 y la impar Xo2 donde Xo2 se comprime utilizando Xe2.

En la compresion sin perdidas predictiva una codificacion eficiente de la prediccion del error:



juega un rol importante. El metodo de prediccion propuesto suele generar errores de prediccion pequeños por el RCT y esquema de prediccion pero aun quedan errores muy altos cerca de los bordes o en las regiones con textura que degradan el rendimiento de compresion.

Para una compresion eficiente las estadisticas de simbolos (errores de prediccion) se deben describir bien con los parametros y/o modelos apropiados. Modelamos la prediccion de errores como una variable aleatoria



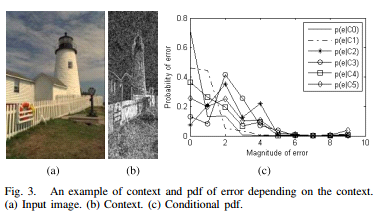
Donde Cn es el contexto de codificacion que refleja la magnitud de los bordes o las texturas. Cn es el nivel de pasos de cuantizacion de la actividad de pixels definida como:



La actividad local y sus pasos de cuanqitzacion se calculan con pixels de Xe porque estos estan disponibles y sus propiedades estadisticas son las mismas que para Xo. La actividad local se cuantiza en K pasos de forma que Cn representa el paso



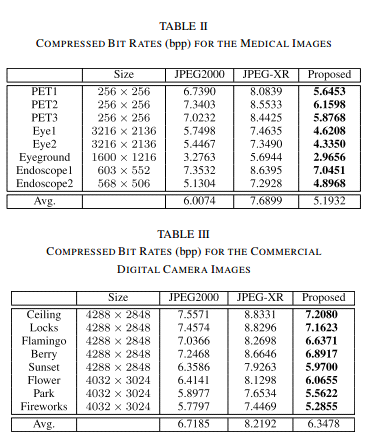
para n=1..K con q0=0 y qk infinito. El tamaño de los pasos de cuantizacion se determina de forma que cada uno incluye el mismo numero de elementos (actividades locales). Para cada contexto se utiliza un codificador adaptativo aritmetico para codificar la prediccion del error.



Se describe la propiedad estadistica de error bastante bien, la magnitud de error es grande cuando la actividad local es fuerte. El modelo propuesto puede ser efectivo para compresion con codificacion aritmetica.

**Resultados experimentales**

El metodo CALIC da mayor compresion que JPEG-LS a costa de una mayor complejidad computacional. Para la compresion en color JPEG2000 y JPEG-XR son mejores que la compresion por canales de CALIC incluso despues de RCT. Comparamos el metodo propuesto con JPEG2000 y JPEG-XR.



**Designing an Efficient Image**

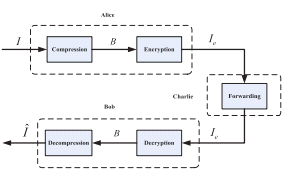
**Encryption-Then-Compression System via Prediction**

**Error Clustering and Random Permutation**

**Abstract:** Por lo general la ecnriptacion se hace antes de la compresion de imagen. Esto supone un problema de como diseñar los algortimos de encriptacion y compresion ya que la compresion de los datos encriptados no es eficiente. Se diseña un sistema de encriptacion y compresion (ETC) donde la compresion con perdidas y sin ellas se tienen en cuenta. Ek esquema opera en el dominio de prediccion de error y da un nivel de seguridad alto. Tambien se demuestra que la aproximacion de codificacion aritmetica se puede explotar para comprimir eficientemente las imagenes. La compresion propuesta es un poco peor en terminos de compresion que los codificadores de imagen que ya hay que toman la imagen desencriptada como entrada. En contraste con muchos ETC la eficiencia de compresion no se ve muy afectada.

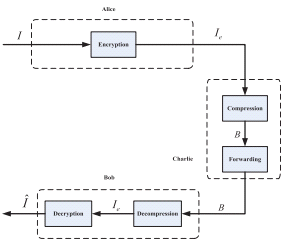
**Introduccion**

Pongamos como ejemplo que Alice quiere enviar una imagen I a Bob por medio de un canal inseguro proveido por Charlie. Lo normal seria comprimir I en B y luego encriptar B en Ie con una funcion de encriptacion Ek donde k es una clave secreta. Los datos encriptados Ie se pasan a Charlie que lo manda a Bob, cuando recibe Ie Bob desencripta y descomprime para recontruir I.



El paradigma de compresion de compresion y encriptado (CTE) tiene ciertos requisitos para muchos escenarios de transmision y el orden de encriptado y compresion se debe invertir en algunas situaciones. Alicia quiere proteger sus datos pero no quieres utilizar sus limitados recursos computacionales comprimiendo una imagen sobretodo cuando esta utilizando un dispositivo con pocos recursos como un movil. En contraste, el proveedor del canal Charlie quiere comprimir todo el trafico de la red para maximizar su uso por lo que seria mejor delegar el trabajo de compresion en Charlie que tiene recursos computacionales abundantes.

Un gran reto para el framework ETC es que la compresion debe ser conducida al dominio encriptado y Charlie no tiene acceso a la clave secreta k.



LA posibilidad de procesar señales encriptadas ha recibido mucha atencion en los ultimos años. Primero se ve que Charlie no va a poder comprimir los datos encriptados ya que no hay ninguna estructura de señal que pueda ser utilizada por un compresor tradicional. Johnson vio que un flujo de datos encriptados se puede comprimir con codificacion con principios de información secundarios sin comprometer la eficiencia de compresion ni la seguridad. Se propusieron unos algoritmos practicos para comprimir sin perdidas imagenes binarias encriptadas. Schonberg investigo despues el problema cuando las estadisticas de fuente subyacentes son desconocidas y las fuentes tienen memoria. Aplicando codigos LDPC en varios planos de bits y explotando la inter/intra correlacion Lazzeretti y Barni presentaron varios metodos para comprimir sin perdidas imagenes encriptadas. Kumar y Makur aplicaron la aproximacion al error de la predicción en el dominio de error y consiguieron una compresión mejor. Ayudado por rate-compatible punctured turbo codes Liu desarrollo un meotodo progresico para comprimir sin perdidas un flujo encriptado de imagenes. Recientemente Klinc extendio el framework de Jhonson para el caso de comprimir bloques de datos encriptados.

Zhang propues un framework de codificacion sin perdidas escalable de imagenes encriptadas via construccion multiresolucion. Un mecanismo compressive sensing (CS) se utilizo para imagenes con encriptacion lineal. Un algoritmo modificado se puede aplicar para estimir la imagen original de los datos encriptados y comprimidos. TAmbien diseño otro encriptado de imagen via la permutacion en el dominio del pixel demostrando que el archivo encriptado se puede comprimir descartando asperezas y la informacion fina de los coeficientes en el dominio transformado. Recientemente sugirio una nueva compresion con descompoisicion multicapa. Se estan desarrollando extensiones para video encriptado.

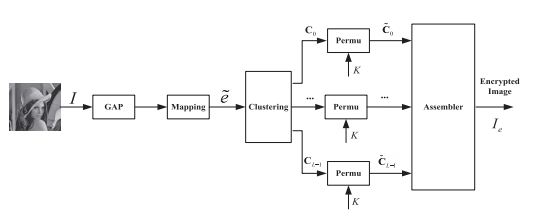
Los sistemas ETC existentes aun tienen una compresion baja comparado con los codificadores para entrada no encriptada. Se quiere diseñar un par enciptacion de imagen y compresion que sea tan eficiente como la compresion sin encriptado.. Se asume que se utilizan imagenes en 8 bits en escala de grises, se propone una aproximacion de basada en la permutacion en el dominio del error. Una codificacion adaptativa al contexto (AC) puede comprimir los datos eficientemente. Hay una penalizacion de compresion <0.1%. Debido a la alta sensibilidad de la secuencia de prediccion de error contra disturbios? hace que se mantenga un alto nivel de seguridad.

**Sistema ETC propuesto**

Alice encripta, Charlie comprime y Bob desencripta y descomprime.

*A. encriptacion de imagen via prediction error clustering and random permutation*

El algortimo de compresion debe considerar la compresion tambien. Se propone un esquema de encriptado operado en el dominio de prediccion de error.



Para cada pixel Ii,j de la imagen I a encriptar, la prediccion se hace con un predictor de imagen GAP o MED. Se adopta el metodo GAP por su capacidad de decorrelacion. El resultado se refina con un mecanismo de feedback adaptativo al contexto. El error se puede calcular como:



Para imagenes de 8 bits el error de prediccion ei,j puede variar entre -2255 y 255, se puede mapear en un rango de 0 a 255 considerando que el valor predicho Iij, esta disponible en el lado del decodificador. Sabemos que el intervalo tiene 256 valores. Si Ii,j<=128 Se pueden reajustar los errores de prediccion posibles para mapearlos en 0 a 255. Si Ii,j>128 se puede aplicar un mapeado similar. Para poder revertir el mapeado es necesario conocer el valor Ii,j. El error de prediccion mapeado va de 0 a 255.

Se dividen los errores de prediccion en L bloques basados en una aproximacion de contexto adaptativa. La aleatorizacion y compresion se benefician de esta operacion. Un estimador de energia de eror se usa como indicador de las actividades locales de la imagen. Para cada pixel el estimador es:



El diseño del cluster debe considerar la compresion y la seguridad. En un proceso offline de pureba se recolectan una serie de samples de las imagenes de prueba. Una tecnica de programacion dinamica se puede emplear para conseguir un cluster optimo en un sentido de la entropia minimo, esta se minimiza.



El termino h denota la entropia de la secuencia de error de prediccion en el cluster y una aproximacion del bitrate que representa todos los errores de prediccion. El cluster tiene una capacidad de compresion optima.

El parametro de seleccion L necesita balancear la seguridad y la complejidad de compresion. Un tamaño mayor da mas seguridad. Se obtiene un valor de 16 heuristicamente. El indice k de cada cluster se saca como:



El algoritmo va asi:

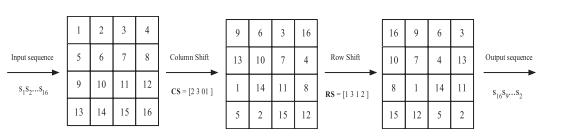
1- Computar todas las predicciones de error de toda la imagen.

2- dividirlos en L clusters. Cada Ck se forma concatenando las predicciones de error mapeadas en orden de raster scan.

3- reshape estos errores en cada Ck en un bloque 2d teniendo 4 columnas y filas. Donde |Ck| es el numero de predicciones de error Ck

4- hacer dos key-driven ciclical shift operations para cada bloque y leer los datos en orden raster-scan para obtener el cluster permutado

CSk y RSk son los vectores de clave secreta que controlan los offsets de columna y fila. Se obtienen de un flujo de llave generado por un cifrador de stream que implica que los vectores pueden ser diferentes para imagenes encriptadas en diferentes sesiones. La permutacion aleatoria:



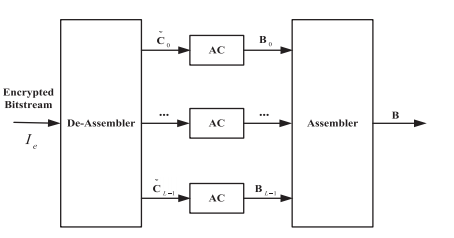
Despues de la permutacion la primera fila es 1,2,3,4 y las segunda 5,6,7,8... LA columna se desplaza por un vector clave CS donde cada columna se somete a un downward cyclical shift de acuerdo con la clave asociada a cada columna. Es proceso se repite para cada fila. Las operaciones de permutacion se hjacen con desplazamientos circulares faciles de implementar.

5- el ensamblador concatena todos los clusters permutados y genera la imagen encriptada. Cada error de prediccion se representa con 8 bits. Su numero es igual que la cantidad de pixels asi que el tamaño de archivo se mantiene

6- se pasan los datos a Charlie. Los valores de Ck hacen que pueda dividir Ie en L clusters.

*Lossless compresion via adaptative AC*

La compresion se hace en el dominio encriptado. Charli no tiene acceso a la clave secreta k.

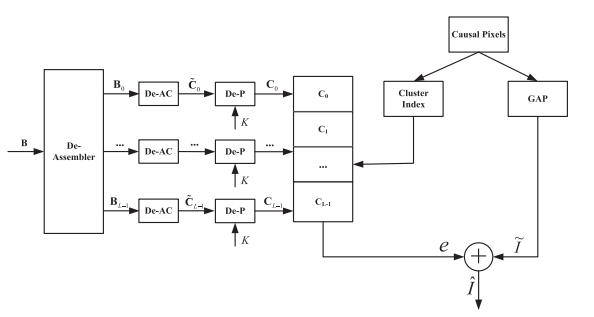


Se utiliza un desensamblador para parsea Ie en L elementos igual que en la parte de encriptacion. Un AC adaptativo se utiliza para condificar sin perdidas cada secuencia de prediccion de error en un stream binario Bk. Esto se puede hacer en paralelo. Un ensamblado concatena todos los Bk para hacer el stream comprimido B.

El tamaño de B se tiene que enviar a Bob. La compresion de cada Ck se basa en que cada permutacion aleatoria solo cambia las posiciones pero no los valores de prediccion de error. ASi se preserva una funcion de probabilidad PMF de la secuencia de predicciopn de error que contrla el AC. El tamaño del flujo se calcula con:



*Sequential decrypt an decompres*



Bob divide B en L segmentos Bk cada uno asociado con un cluster de error de prediccion. Para cada Bk un decoder adaptativo aritmetico se aplica para obtener la secuencia de error permutada. Como Bob sabe la clave K la despermutacion se puede usar para obtener el Ck original.

Con todos los Ck la decodificacion de cada valor de pixel se hace en orden raster-scan. Para cada I,j el estimador de energia de eror y el valor predecido se pueden calcular por redondeos causales que ya han sido decodificados. El primer unused prediction error en el clustrer kth se selecciona y se utiliza para derivar eij de acuerdo a iij y la regla de mapeado descrita antes. Se pone una bandera usado en el error de prediccion procesado. El pixel reconstruido se computa como:



No hay distorsion en en la prediccion del error lo que implica decodificacion sin errores.

*security analysis*

El flujo de llave puede ser diferente si la imagen se encripta en sesiones diferentes. El unico modo de ataque aplicable es ciphertext-only en el que el atacante solo puede acceder al chipertext y probar a recuperar la imagen original.

Como el modulo AC es publico y reversible el atacante puede obtener la imagen encriptada Ie concatenando los L clusters de prediccion de error. Con Ie el siguiente ataque estadistico se puede aplicar. Como el tamaño de Ck es publico Ie se puede partir en L segmentos. Para cada Ck la probabilidad (EPMF) es:



La posibilidad de comprimir la imagen encriptada sin clave da la posibilidad al atacante de aplicar la misma estrategia de compresion y el tamaño del archivo resultante es un indicador estadistico de la imagen. Aun asi es practicamente imposible imaginar la permutacion debido a la gran cantidad de formas de hacerla. Esta cantidad se puede calcular como:



En la practica este numero es enorme. En la imagen de Lena es 2^256.

El atacante puede intentar decodificar el archivo encriptado Ie directamente. Necesita el error de prediccion y el valor de prediccion asociado. Una forma de adivinarlo es estimando el indice k del cluster de acuerdo con los pixels vecinos codificados y seleccionar el elemento de Ck. El indice Ck se determina de acuerdo al estimador de energia de error calculado con los pixels casuales de alrededoer. Esto implica que estos pixels estan decodificados correctamente y cualquier error anterior produce una propagacion del error influyendo a la determinacion correcta del cluster k. La estimacion exacta del valor Iij tambien depende de la reconstruccion corecta de los pixels de alrededor. El efecto de propagacion de error ayuda a mejorar la seguirada. Si se puede estimar k corectamente el atacante tiene que decidir el valor eij respecto a Ck. Como la distribucion de errores de prediccion esta tiene picos de cerola estimacion optima es cero. Poniendo todos los valores a cero no dan como resultado la imagen.

Otra forma es explorar la correlacion de imagenes naturales cuando se estiman los errores de prediccion. Suponmgamos que el atacante puede estimar los pixels hasta Iij. El atacante necesita seleccionar una prediccion de error del cluster. Debido a la correlacion espacial el atacante puede deshechar esta seleccion. La correlacion espacial ayuda al atacante. Suponemos un escenario general. Si el atacante consigue un 99% de los errores de prediccionde las dos primeras filas. Se supone que los errores de prediccion estan disponibles. Los valores de PSNR son muy bajos 10db. La permutacion propuesta sigue sirviendo mientras no se necesite un secreto total.

*compression performance*

Los predictores GAP y MED tienen una decorrelacion fuerte. Hay muy poca interdependencia en la secuencia de prediccion de error. Comparado con la codificacion predictiva en la que la compresion se conduce sobre las predicciones de error sin permutar esta se hace sobre las permutadas.

Una secuencia con interdependencia, causada por redundancia es mas compresible que su opuesta. Las operaciones de permutacion en el dominio de prediccion de error destruyen la interdependencia. El resultado se comprime menos que la version sin permutar. Afortunadamente la interdependencia que queda es limitada gracias a la decorrelacion superior de los predictores de imagen. Se podra ver que la penalizacion de compresion es baja.,

***resultados experimentales***

El mono se ve mas claro porque tiene mas zonas con texturas que son mas dificiles de comprimir y dan valores de error de prediccion mayores.