هذا الملحق شامل لأفكار الدكتور مدحت ومساعد لفهما وتسلسها ولكن لا يجب اتخاذه كمصدر أساسي لدراسة المادة

الصوت - Sound

الصوت: هو موجة ميكانيكية لا تنتقل عبر الفراغ بعكس الضوء (حيث ان الضوء موجة كهرطيسية).

منشأ الصوت:

ينشأ الصوت عن الأجسام المهتزة ضمن وسط ما مثل الهواء.

تمثيل الصوت:

يتم تمثيل الصوت بالزمن والمطال (أو الشدة) (يمثل كـ Frequency).

الصوت عبارة عن إشارة Analog، نحتاج الى تحويلها إلى Digital؛ لأننا بحاجة لها بالصيغة الرقمية من أجل المعالجة الحاسوبية (تشفير، تخزين، ...)، وتدعى هذه العملية بالـ Digitizing، كما نحتاج الى العملية العكسية (أي التحويل من Digital إلى Analog).

التقطيع - Sampling

هي عملية مقابلة كل فولط كهربائي بقيمة Integer، ونقوم بالتقطيع من أجل الاحتفاظ بعينات من الإشارة لأنه من الصعب تخزين منحني الإشارة كاملا، حيث يتم اختيار عينة كل فترة زمنية محددة pt (يجب معرفة كم عينة نحتاج اليها في الثانية).

التقطيع على محور x يعتمد على ال frequency ويؤثر على هوية الصوت (يمكن من خلال التقطيع ان يتحول صوت الرجل الى صوت طفل مثلا).

- نزيد عدد العينات في حال كانت الجودة مطلوبة وبالتالي يكون حجم الملف كبير.
- نقلل عدد العينات في حال كانت السرعة مطلوبة وبالتالي يكون حجم الملف أصغر.

معايير Sound Digitizing

ذكرنا سابقاً أن عملية الـ Digitizing هي عملية تحويل الإشارة الصوتية من Analog إلى Digital.

1 – معدل التقطيع Sampling Rate:

عدد العينات التي يتم تقطيعها في الثانية ويقاس بالـ (Hz).

2 – عمق التقطيع (Resolution – Sound Depth – Bit Depth – Rate):

عدد البتات المستخدمة لتمثيل كل عينة ناتجة عن التقطيع بعد تكميمها.

التكميم: هي عملية تقريب القيم الناتجة عن التقطيع إلى قيم معينة من أجل تخزينها.

من خلالها يتم التوزيع قيم العينات على مجال من العينات وحسب قيم المجال نحدد العمق

حيث أن عملية التكميم هي تقطيع على المحور y.

$$\text{عدد مستويات التكميم} = 2^{\text{عمق التقطيع}}$$

كلما كان العمق أكبر كان الصوت بدقة أكبر

خطأ التكميم: هو الفرق بين الإشارة التماثلية الأصلية والإشارة بعد التكميم.

3 – **عدد القنوات** Number of Channels (عدد إشارات الصوت المطلوبة):

عدد القنوات التي ينتقل الصوت من خلالها. أنواعها:

A) Monophonic (Mono):

نقل الصوت عن طريق قناة واحدة، ميكروفون واحد للتسجيل، وجهاز خرج واحد.

B) Stereophonic (Stereo):

يتم نقل الصوت عبر قناتين مستقلتين (كما في سماعات الجوال).

C) Quadraphonic:

يتم النقل عبر 4 قنوات مختلفة (زوايا).

D) Sorrund Sound:

6 قنوات (التسجيل: مسجلين أماميين، جانبيين، خلفيين).

E) 5.1 Multi Channel:

5 قنوات. 3 امامية، 1 منتصف، 1 يمين، 1 يسار وقناة إضافية تدعى بالـ LFE (Low Frequency Extension).

F) 6.1 Multi Channel:

6 قنوات (قناة خلفية زيادة على الـ 5.1).

نتيجة: بزيادة عمق التقطيع ومعدل التقطيع وعدد القنوات تزداد جودة الصوت ودقته.

يكون التقطيع اقل ما يمكن في الهواتف العادية telephone واكثر ما يمكن في ال DVD.

حساب حجم ملف:

$$File\ Size = \frac{Sampling\ Rate \times Resolution \times channels \times time\ (s)}{8}$$

حالة Mono: نقوم بالضرب بـ 1.

$$File\ Size = \frac{Sampling\ Rate \times Resolution \times 1 \times time\ (s)}{8}$$

حالة Stereo: نقوم بالضرب بـ 2.

$$File\ Size = \frac{Sampling\ Rate \times Resolution \times 2 \times time\ (s)}{8}$$

يمكن تقسيم الناتج في كلا الحالتين بـ 2^{20} للحصول على الحجم بالـ MB مثلا.

استراتيجيات تخفيض الحجم

تقليل زمن الملف

تقليل عدد القنوات: يؤثر على ال effects للصوت (صوت المحيط).

تقليل معدل التقطيع (frequency): يؤثر على هوية الصوت

تقليل عمق التقطيع (Bit rate): يؤثر على جودة الصوت (يصبح الصوت مشوش) دون التأثير على هوية الصوت.

تطبيق خوارزميات الضغط: loosing – lossless.

تردد الأصوات Frequency

حساسية أذن الإنسان: $20\ Hz \rightarrow 22\ KHz$

صوت الإنسان: $500\ Hz \rightarrow 2\ Khz$

معدل التقطيع

للتقطيع بمستوى جيد يجب معرفة محتوى الإشارة، كما اننا نهتم بالمجال الذي يستطيع الانسان سماعه.

معدل التقطيع = أعظم تردد (اخر x لها y أكبر من الصفر) × 2

وبالتالي نحصل على عدد العينات التي نحتاجها في الثانية الواحدة (إذا تم اخذ عينات اقل من المعدل تخرب الإشارة، اما في حال اخذ عينات أكثر تكون الدقة أكبر).

صيغ ملفات الصوت – Audio File Formats

1 – AIFF (Audio Interchange File Format).

تم تطويره من قبل Apple، لا يضغط الملفات، ويتم تخزين العينات على 8 بت أو 16 بت.

2 – WAV (Waveform Audio File Format)

لدى Microsoft، يدعم الضغط وعدمه، يتم تخزين العينات أيضاً على 8 بت أو 16 بت.

3 – AV: 8 بت – بطيئة بالضغط.

4 – RM: ضغط كبير جداً.

5 – MP3 (MPEG Audio Layer 3): ضغط بمعدل جيد، جودة عالية للصوت.

$$1\ HZ = 1\ Cycle\ /\ Second$$

FFT – Fourier Transformations

تحويل فورييه: هو إيجاد مجموعة من الإشارات التي ترسم Sin/Cos وعند جمعها نحصل على الإشارة الأصلية.

لكل إشارة جزئية هناك Frequency، كلما زاد هذا الـ Frequency يكون عدد تكرارات الشكل أكبر.

ولكل جزء هناك صفحة إزاحة ومطال.

الإشارة الاصلية ليست دورية لذلك نفرض أنها دورية (نقوم بتكرارها).

من أجل تحويلها إلى مجموعة اشارت دورية (Sin / Cos):

$$F(f) = \int_{-\infty}^{\infty} f(t)\, e^{-2\pi jft}\ dt$$

عندما تكون الإشارة مستمرة يكون المجموع هو تكامل.

العلاقة العكسية: يمكننا القيام بالتحويل العكسي من فضاء ال frequency الى فضاء الإشارة:

$$f(t) = \int_{-\infty}^{\infty} F(f)\, e^{2\pi jft}\ df$$

حيث F(f) هو عدد عقدي (كل قيمة منه عدد عقدي).

لو كانت الإشارة بالأصل هي Sin فعند تحويلها تصبح نبضة ديراك في فضاء ال frequency.

مساهمة كل Frequency هي عدد عقدي (حيث نقوم خلال الرسم برسم طويلة العدد العقدي).

حالة خاصة: إذا كانت قيمة ال frequency تساوي الصفر تكون الإشارة خط مستقيم.

الإشارات المتقطعة – DFT (Discrete Fourier Transformation)

عند تحويل الإشارة المستمرة بتحويل فورييه ينتج لنا منحني فيه y عند Max، و $x = 0$ أكبر مساهمة، وشكل متناظر حيث مساهمة الباقي هو الصفر.

ولكن عند الإشارات المقطعة يكون هناك تكرار للإشارة وبالتالي تتكرر نقطة المساهمة فنأخذ أكبر frequency ونقطع بضعفه، أي اعلى frequency يكون $\frac{f_s}{2}$ ويصبح المجال $\left[-\frac{f_s}{2}, \frac{f_s}{2}\right]$

$$F(f) = \sum f(t)\, e^{-2\pi j f t}\, dt$$

عند التقطيع بـ f_s (أي freq) تكون الأطراف هي $\left[-\frac{f_s}{2}, \frac{f_s}{2}\right]$ (لانه في التقطيع مضروب بـ 2)

مثال: بفرض لدينا 256 عينة:

العينة 1 هي مساهمة أول نقطة وقيمة freq = $\frac{f_s}{256}$

العينة 2 هي مساهمة ثاني نقطة وقيمة freq = $\frac{2f_s}{256}$

عدد العينات يجب أن يكون = 2^n

ملاحظة:

عند تحويل الإشارة المتقطعة بفورييه وعندما نأخذ $t = 0$ تعطي قيمة للقمة عالية جداً لا نستطيع رسمها لذلك نرسم بـ log الطويلة.

الهدف من تحويل فورييه:

فضاء الـ Frequency يفيد بمعرفة هوية المتحدث. بحيث يوجد لكل شخص مجال frequency معين.

- اول قمة بعد الصفر هي f_0 تحدد عمر الشخص مثلاً.
- ال f الخاصة بالطفل تكون عالية جدا (لان صوته يزقزق زقزقة).
- كلما اخذنا f_i أكثر حددنا هوية الشخص بشكل أدق.
- الصوت القوي جداً يعطي انطباع فيغطي على بقية الأصوات حتى زواله.

تحويل فورييه للصورة

الصوت هو إشارة أحادية البعد أما الصورة هي إشارة ثنائية البعد (Grayscale).

2D: محورين x و y وارتفاع (قيمة الإضاءة المخزنة بكل بكسل).

عند تحويل فورييه نحتاج لتركيبها الى معادلة ذات بعدين:

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M}\sum_{y=0}^{N} f(x,y)\, e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)}$$

حيث: M عدد العينات الكلي.

يرد التابع السابق عدد عقدي عند رسم مطاله ينتج لنا رسمة أخرى (ويمكن رسم log الطويلة).

هنا النتيجة هي مجموع إشارات Sinc x:

$$Sinc\ x = \frac{Sin(x)}{x}$$

F تمثل عن طريق Magnitude و Phase بدل من Real و Imaginary

$$Magnitude(F) = \sqrt{real(f)^2 + imaginary(f)^2}$$

$$Phase(F) = Atan\left(\frac{imaginary(F)}{real(F)}\right)$$

الهدف من تحويل فورييه على الصورة:

كل مالا نستطيع القيام به في الفضاء t نستطيع القيام به في الفضاء frequency.

مثل:

التعديل: عندما نرسم فورييه نستطيع تحديد الحواف (الـ f العالي هو تغير حدي بالإشارة)، التغير السريع (الانتقال بين لونين) يخلق f عالي، الـ f العالية تمثل الحواف.

عندما نريد تطبيق تغبيش على الصورة (تطبيق فلتر Blur) نقوم بأخذ البكسلات ال 8 المحيطة بالبيكسل المراد تغبيشه وحساب متوسطها الحسابي (كيف تتم هذه العملية في فضاء ال frequency؟)

نقوم بتحويل الصورة باستخدام فورييه ونلاحظ من رسم الإشارة وجود قمة، في حال النزول بسرعة (انتقال حاد بين لونين) يتم خلق freq عالي أي أصبح هناك مساهمة عالية لإشارة ال sin (الـ freq الخاص بها عالي ويوجد فرق في اللون) (نقوم فقط بحذف الاحداثيات الكبيرة ذات الـ f العالية) (التخلص من الحواف) (فكرة الضغط).

عند الضغط بهذه الطريقة نخزن مساهمات كل frequency وهذا يقلل الحجم ولكن عند فتح الصورة علينا تحويلها بتحويل فورييه العكسي حتى نستطيع قراءتها، وتعود الى الحجم الأصلي ولكن بقيم مختلفة.

ملاحظة: لا نقوم بتحويل فورييه للصورة الكبيرة (لأنها ستأخذ وقتا كبيراً)، ولا الأصوات الكبيرة لأنها سوف تقوم بالتقاط أي صوت لسنا بحاجة اليه.

Image Compression

هناك نوعين للضغط:

1 – Lossy: نفقد جزء من البيانات، وتكون درجة الضغط فيها عالية.

2 – Lossless: ارجاع نفس البيانات دون فقدان شيء، وتكون درجة الضغط فيها ليست عالية.

RLE (Lossless)

تعتمد على تتالي التكرارات، أكبر فائدة لها هي في الصور الـ Binary (0/1) (الأبيض والأسود).

كيف يتم ضغط صورة ال grayscale؟

1. التحويل الى binary.
2. الضغط باستخدام RLE.

عند تحويل الأرقام العشرية الى ثنائية قد نقع في مشكلة عدم تتالي التكرارات مما يسبب عدم فعالية هذه الخوارزمية.

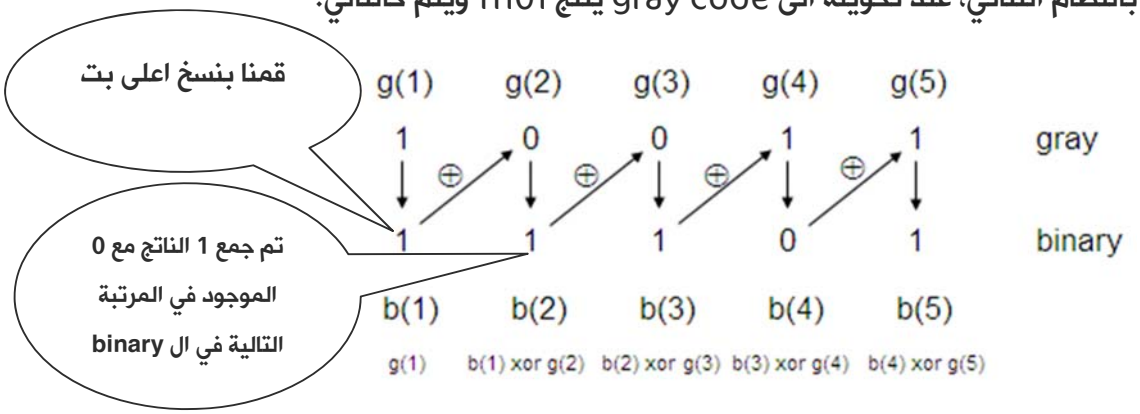
وبالتالي لضغط صورة grayscale نقوم بما يلي:

1. نحول الى binary
2. التحويل من binary الى gray code: وتتم كما يلي:

a. نضع اعلى قيمة في ال gray code نفسها التي في ال binary.

b. نجمع (XOR) كل بت ناتج من ال gray code مع البت التالي في ال binary.

ليكن لدينا العدد 10011 بالنظام الثنائي، عند تحويله الى gray code ينتج 11101 ويتم كالتالي:



تذكرة: بوابة XOR هي بوابة عدم التماثل خرجها 1 في حال كان البتين مختلفين وإلا 0 في حالة التماثل

Statistical Methods (Entropy)

نحسب تكرار كل رقم والأرقام المكررة كثيراً نخزنها بعدد بتات أقل، وهي مناسبة لصور ال gray scal والهدف منها التخلص من التكرار.

لا تعد خياراً جيداً لضغط الصور! بحيث نهتم بالإضاءة أكثر من اللون.

مثال عليها: **Entropy encoding**: وتعرف كالتالي:

Entropy encoding: technique that replace data element with code representation.

PCM (Pulse Code Modulation) (Lossless)

تقطيع على الـ x وتكميم على الـ y.

تذكرة: التكميم هي عملية تقريب القيم الناتجة عن التقطيع إلى قيم معينة من أجل تخزينها.

في حالة المسافات الثابتة على المحور y نقوم بتسمية هذا النوع بـ LPCM (Linear PCM).

حيث نحن ملزمين بأخذ مسافات ثابتة على الـ x ولكن لسنا ملزمين بأخذ مسافات ثابتة على الـ y.

DPCM (Differential Pulse Code Modulation)

يهمنا الفروقات بين العينات (نخزن الفرق).

مزاياها:

- قيم الفروقات أقل حجماً من قيم العينات (Decode by less bits).
- تكرار كبير لقيم الفروقات (High Compression Rate).

كما يمكن أيضاً أن نخزن الفرق بين القيمة التي نتنبأ بها وبين القيمة الصحيحة (فرق الخطأ) التنبؤ من خلال القيم السابقة (نكون بحاجته في خوارزميات التوقع).

إلى الآن كانت الخوارزميات Lossless حتى نطبق الـ Lossy نقوم بتكميم ناتج ما سبق DPCM, PCM

فمثلاً: يحتاج مجال الفروقات للتخزين على 7 بتات فنقوم بتخزينها على 4 بت وبذلك نقوم قد خسرنا جزء من البيانات ولا يمكن استعادتها.

Transform

تقوم بالتحويل من فضاء إلى فضاء آخر، بحيث n عينة من الفضاء القديم تبقى n عينة في الفضاء الجديد.

الفائدة: العينات في الفضاء الجديد ستكون مترابطة بشكل أقل مما يسمح بضغطها بفعالية أكبر.

Image Frequency

تقاس بعدد الألوان المتغيرة على طول السطر.

Freq تعني الحافة (حافة تفصل اسود عن أبيض مثلاً) تكون ذات freq عالي.

Low Frequency هو الأكثر أهمية (اساسيات الصورة) تكميمه خفيف (عدد بتات أقل)، وبالتالي ضغط قليل.

High Frequency هو الأقل أهمية (تفاصيل الصورة) تكميم عالي، وبالتالي ضغط كبير.

بدلاً من الـ FFT نقوم بالـ DCT (يستخدم للإشارة المتقطعة).

DCT (Discrete Cosine Transformation)

DCT on 1D:

1D مثلاً الصوت، فيه N عينة قيم y هي f(i) (نريد كتابته بدلالة Cos لكن دون عقدية).

$$F(j) = \sqrt{\frac{2}{N}}\, C(j) \sum_{i=0}^{N-1} f(i) \cos\left(\frac{\pi j(2i+1)}{2N}\right)$$

حيث:

$$C(j) = \begin{cases} \frac{1}{\sqrt{2}} & j = 0 \\ 1 & j > 0 \end{cases}$$

التحويل العكسي:

$$f(j) = \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} C(j)\, F(i) \cos\left(\frac{\pi j(2i+1)}{2N}\right)$$

الحد الأعلى F (0) اسمه DC والبقية يسمون AC.

DCT مشابه لـ DFT الفرق أنه يعطي أعداد حقيقية بينما DFT يعطي اعداد عقدية.

نتائج

FT: تحويل فورييه لإشارة مستمرة (غير مستخدم).

DFT: تحويل فورييه لإشارة متقطعة (N عينة ترد N عينة).

FFT: هي DFT نفسها ولكن أسرع (أفضل للحاسوب).

DCT: هي DFT نفسها ولكن يرد أرقام حقيقية وليست عقدية.

بمقارنة رسم DCT وDFT نجد أن DCT الإشارة فيها مركزة أكثر (طاقة الإشارة مركزة)

DCT on 2D:

$$F(u,v) = \left(\frac{2c(u)c(v)}{N}\right)\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} f(m,n)\cos\left(\frac{2m+1}{2N}u\pi\right)\cos\left(\frac{2n+1}{2N}v\pi\right)$$

$$u = 0,1,\dots,N-1 \quad v = 0,1,\dots,N-1$$

حيث:

$$c(k) = \frac{1}{\sqrt{2}} \quad for \quad k = 0$$

$$1 \; Otherwise$$

التحويل العكسي:

$$f(m,n) = \frac{2}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} c(u)\, c(v)\, F(u,v)\cos\left(\frac{2m+1}{2N}u\pi\right)\cos\left(\frac{2n+1}{2N}v\pi\right)$$

$$u = 0,1,\dots,N-1 \quad v = 0,1,\dots,N-1$$

عندما تكون

$$u = 0 \; and \; v = 0 \Rightarrow \sum\sum F(u,v)$$

وهي DC.

JPEG Compression

يمكن أن تكون Lossy أو Lossless. ونستطيع التحكم بمقدار الخسارة التي نريدها.

خطوات الضغط:

1 – تحويل من RBG إلى نظام لوني آخر يدعم الـ إضاءة/لون (luminance/chrominance) وذلك لأن العين حساسة للتغيرات الصغيرة للإضاءة أكثر من اللون مما يسمح بضغط كبير للون.

تعتبر هذه الخطوة خطوة اختيارية، ولا يتم تطبيقها على الـ grayscale images.

2 – الأجزاء الخاصة باللون Chrominance يتم القطع منها.

أي ننشأ حجم جديد للصورة (غالباً يكون 4:1:1 أو 4:2:2).

يتم تطبيقها فقط لأجزاء التلوين Chrominance وليس للإضاءة Luminance.

☒ لا تستخدم لـ Grayscale images.

تسمى عملية sub/down sampling.

3 – ننشأ blocks من 8*8 بكسل (Data unit)

في حال لم تناسب التقطيع نقوم بتكرار الأسطر والأعمدة حتى يصبح العدد من مضاعفات 8.

ويوجد لدينا نمطين للتعامل:

Non-interleaved mode

هنا نأخذ كل كتل النسخة الأولى، ثم كل كتل النسخة الثانية، فالثالثة وهكذا.

Interleaved mode

هنا نأخذ جزء من الأولى، وجزء من الثانية، وهكذا.

4 – نطبق DCT على الـ Data Unit (Blocks):

بما أن DCT تتعامل مع cos بالتالي لها مجال محدد من القيم، هذا يعني أنه سوف يكون هناك ضياع في المعطيات.

DC يقوم بقياس متوسط طاقة الـ Block.

الطاقة سوف تتركز في الزاوية العليا اليسرى.

5 – التكميم – Quantization (هنا تكمن مرحلة الخسارة):

خطأ التكميم هو المصدر الرئيسي (Main Source) لـ Lossy Compression.

كل تردد من الـ 64 تردد الموجود في الـ Data Unit يقسم على عدد منفصل يدعى بـ QC (Quantization Coefficient)

تقريب الناتج هو Integer.

لدينا: QC(u,v) Uniform Quantization: هو ثابت.

الـ Non-uniform quantization هو أعلى قيمة من اليسار والتي تملك أهمية عالية لذا تملك QC صغيرة.

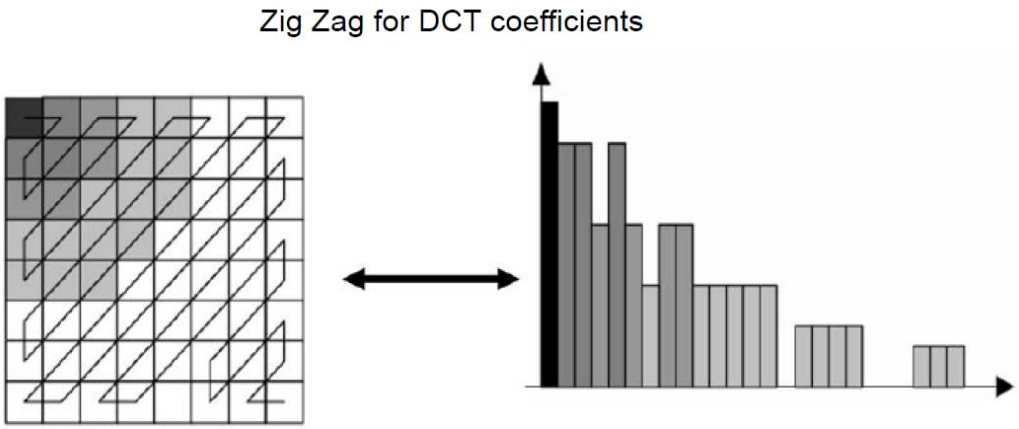
$$F'(u,v) = Round\left(\frac{F(u,v)}{Q(u,v)}\right)$$

$$F(u,v)\ original\ DCT\ coefficient$$

$$F'(u,v)\ DCT\ coefficient\ after\ quantization$$

$$Q(u,v)\ Quantization\ Value$$

6 – تخزين باستخدام الية Zig Zag من خلالها نحصل على شعاع.



7 – Coding:

لـ DC نطبق DPCM.

DC تكون قيمة كبيرة ومختلفة عن باقي القيم ضمن نفس الـ Block ولكنها تكون قريبة من قيم DC للكتل الأخرى لذلك تفيدنا الـ DPCM هنا.

8 – Entropy Coding:

نطبق Entropy على ناتج DPCM حيث تخزن بالشكل (size, value).

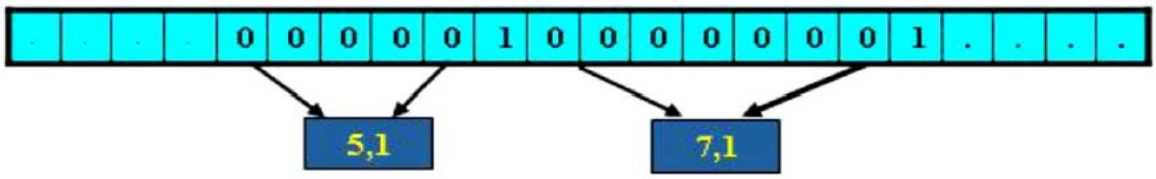
Size: عدد البتات اللازمة، Value: البتات.

9 – RLE on AC Components:

بعد الـ Zig Zag نطبق RLE على AC Components حيث تكون بالشكل: (skip, value).

Skip: عدد الاصفار.

Value: أول عدد تالي لا يساوي الصفر.



10 – Entropy Coding on RLE on AC Components:

نطبق Entropy على ناتج RLE تخزن بالشكل (S1, S2).

S1:

- Run Length: طول الاصفار
- Size: عدد البتات اللازمة لتخزين اول رقم تالي غير الصفر

S2:

قيمة الـ AC Component.

أنماط JPEG

1 – Sequential Mode: يتم القيام بعملية (Decode, Encode) بـ Single-run.

2 – Progressive: العناصر المهمة أولاً، ولدينا حلان:

A- إرسال DC (Spectral Selection).

B- إرسال اول بت (Successive Approximation)

C- Hierarchical (بديل Progressive) ، إرسال صورة صغيرة، ثم أكبر وهكذا.

Lossless Compression

- $Statistical$: $(RLE, Shannon, Huffman, Adaptive\ Huffman, Arithmetic)$.

المكرر كثيراً يأخذ عدد بتات قليل.

- $Dictionary$: $(LZ77, LZW)$.

$$Compression\ ratio\ = 1 - (B_1/B_0)$$

حيث:

B_0: عدد البتات قبل الضغط.

B_1: عدد البتات بعد الضغط.

ضغط البيانات يدخل مجال نظرية المعلومات بسبب اهتمامه بالتكرار.

في حال تخلصنا من التكرار نكون قد قللنا الحجم.

Information of Symbol:

الـ $Symbol$ هو كل ما أحاول ضغطه.

أهميته في معلوماته $(info)$

$$\inf(a_i) = -\log P(a_i)$$

حيث:

- ضربنا بسالب ليكون الـ inf موجب.
- a_i احتمال فهو تحت الـ 1.

الـ $Symbol$ المكرر كثيراً (احتماله كبير) ليست له أهمية.

Entropy:

هي متوسط وزن المعلومات لكل $Symbol$ (نثقل كل $info$ بالاحتمال)، تمثل متوسط عدد البتات للمعلومات الموجودة في الرسالة.

نتيجتها تمثل الحد الأدنى لعدد البتات من أجل تمثيل الحد الأدنى (وسطياً) لعدد البتات من أجل تمثيل كل $Symbol$

$$(\text{bits/symbol})\ n = -\Sigma_{i=1}^{n} P(a_i) \log_2 P(a_i)$$

$entropy$ لـ $data$ هي مجموعة $entropies$ لكل $symbol$

$$\mu = \Sigma_1^N P(a_i) \log_2 \frac{1}{P(a_i)}$$

Statistical Techniques

Shannon-Fano Coding

الخطوات:

1. نحسب احتمال ورود كل رمز.
2. نفرز الاحتمالات تنازلياً.
3. نقسم الرموز إلى قسمين بحيث يكون احتمال أول قسم تقريباً يساوي احتمال ثاني قسم.
4. إسناد قيمة 1 لأول قسم، 0 لثاني قسم.
5. نكرر العملية.

مثال1:

a1	0.25	1	1				11
a2	0.20	1	0				10
a3	0.15	0		1	1		011
a4	0.15	0		1	0		010
a5	0.10	0		0		1	001
a6	0.10	0		0		0	0001

مثال2:

a1	0.25	1	1		:11
a2	0.25	1	0		:10
a3	0.125	0	1	1	:011
a4	0.125	0	1	0	:010
a5	0.125	0	0	1	:001
a6	0.125	0	0	0	:000

Huffman Coding

الخطوات:

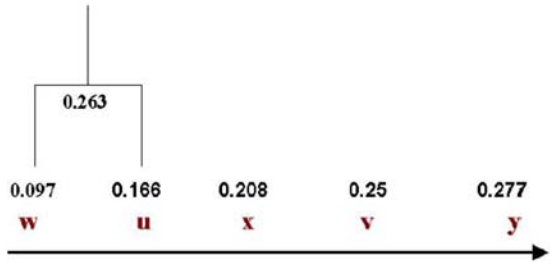
1. نحسب الاحتمال.
2. نرسم شجرة ثنائية $Huffman\ Tree\ (bottom-up)$.
   a. نرتب الاحتمالات تصاعدياً من اليسار إلى اليمين.
   b. أصغر عقدتين (أقصى اليسار) نأخذهم ونشكل منهم عقدة (نجمع احتمال الورقتين ليكون احتمال العقدة الجديدة).
   c. نكرر.
3. نعطي 0 لليسار، 1 لليمين.

مثال:

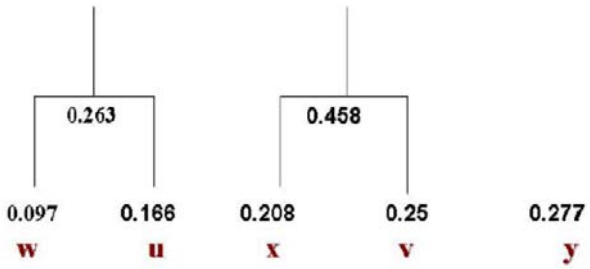
Symbol	Occurrence	Probability	Data size
"w"	7	0.097	56 bits
"u"	12	0.166	96 bits
"x"	15	0.208	120 bits
"v"	18	0.25	144 bits
"y"	20	0.277	160 bits
sum	72	1	576 bits

حجم الرسالة الجديدة هو عدد مرات تكرار الرمز x، عدد البتات ممثل عليه الرمز أثناء تشكيل العقد الجديدة من الممكن أن نضطر إلى إعادة ترتيب الشجرة (مثل المثال) المهم نحافظ على الترتيب التصاعدي الصغير أقصى اليسار.

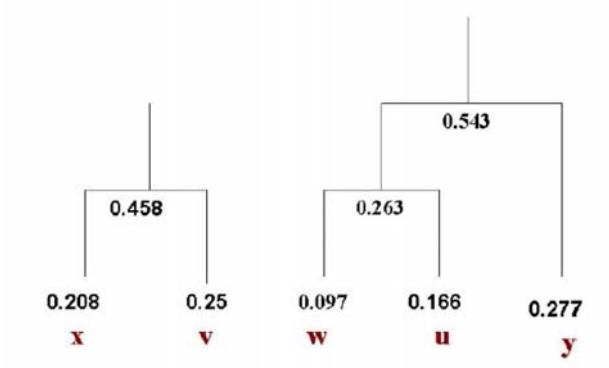
أولاً:



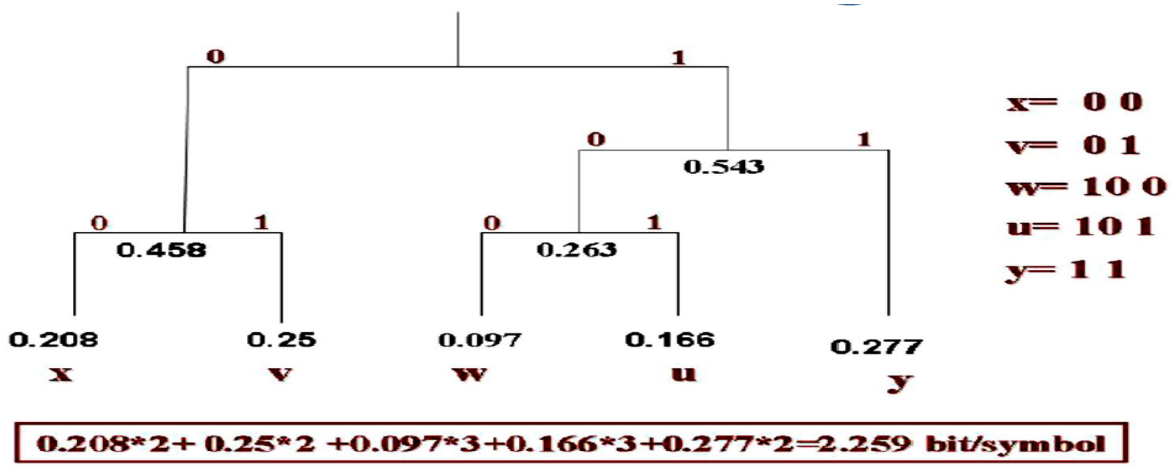
ثانياً:



ثالثاً:



رابعاً:



0.208*2+ 0.25*2 +0.097*3+0.166*3+0.277*2=2.259 bit/symbol

مشاكل *Huffman*

عند فك الضغط نحتاج إرسال الاحتمالات (في الـ *header* يرسل) أو إرسال كامل شجرة هوفمان وبذلك سيختلف حجم الملف المضغوط (سيزيد)

ميزات *Huffman*

- ضغط بدون غموض.
- $n \leq avg\ code\ length \leq n + 1$.

Adaptive Huffman Coding

مشكلة Huffman أنها تحتاج إحصائيات وأيضاً إرسال المعلومات من أجل الفك يشكل مشكلة أيضاً ممكن يكون الـ $stream$ $life$ مباشر سنضطر لانتظار انتهاء ورود المعلومات حتى نحسب الاحصائيات وذلك مشكلة أيضاً،

لحل كل هذه المشاكل ظهرت $Adaptive$.

من يقوم بالضغط ومن يفك، يضغطون بشكل مباشر (على الهواء) حيث لا يوجد إحصائيات وإذا احتجت تستطيع تشكيل إحصائيات لكل قطعة تصلك.

الخطوات: (الفك نفسها عدا الـ $encode$ تصبح $decode$)

1. قراءة $Symbol$ الدخل.
2. تشفير الـ $Symbol$ ($encode$).
3. $update$ للمحتوى.
4. نستعمل الموديل الجديد من أجل ضغط الرمز التالي.

كلما جاء الرمز مجدداً سيخزن على بتات أقل

الآلية

يصل الرمز الأول يتم ضغطه، وفكه معاً ثم يتم تحديث الشجرة عند كلا الطرفين (فتصبح نفس الشجرة عند كل منها)

1. **تهيئة:**

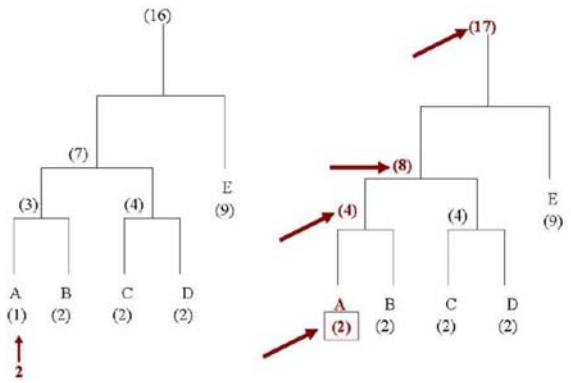
أول رمز سيكتب بدون أي ضغط وثم سينضاف إلى الشجرة ويسند له الكود الخاص به من أجل استعماله عند وروده مرة أخرى.

2. **تجديد الشجرة:**

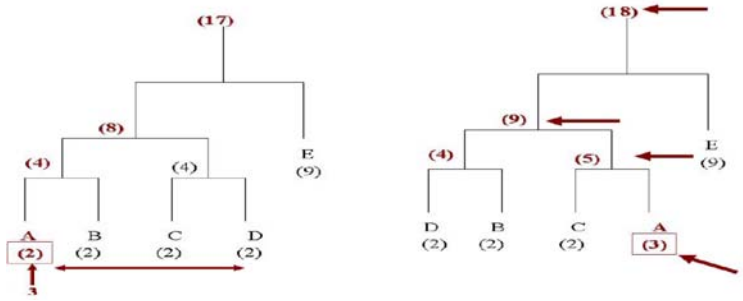
زيادة عدد الـ $freq$ للرمز (ممكن أن نعزل مكان العقدة في الشجرة من أجل المحافظة على الرموز الأقل تكراراً في الأوراق) لم يعد هناك احتمالات بل $freq$

مثال:

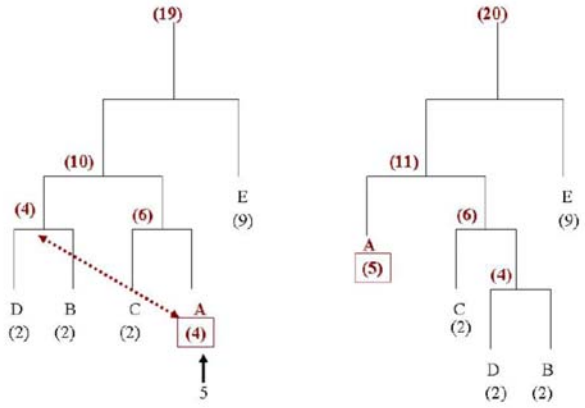
أولاً:



ثانياً:



ثالثاً:



$swap$ من العقدة الحالية وكامل العقدة الأصغر فيها بـ 1.

عند الاستقبال تحتاج لمعرفة نوع الـ $Symbol$ (مضغوط أم لا)

لأن أول إرسال(الأوراق) سترسل المحرف كما هو بالاسكي مثلا

لحل ذلك سنرسل رمز $escape$ قبل كل $symbol$، هذا الرمز $escape$ يحتاج لترميزه أيضاً

$\Longleftarrow$ نضيفه إلى الشجرة (New) يكون لها أطول ترميز وهو ورقة لها $freq = 0$

مشاكل ***Adaptive***

ليس من الضروري أن نصل لأصغر تمثيل لعدد البنات.

Arithmetic Coding

ترميز كل رسالة على رقم عشري $float$ يكون $[0,1[$.

الخطوات:

لكل حرف نأخذ $freq$ واحتمال وروده ثم نشكل $range$:

$$[start, end[$$

حيث:

$$start: low\_range. end: high\ range$$

$$end = (\text{نهاية السابقة})\ start + probability$$

$start = end$ التالي

من أجل أول محرف نضع:

$$low = 0, high = 1, range = 1$$

فيكون:

$$low = low + range * (Start\ \text{الـ})\ low\_range(s)$$
$$high = low + range * high\_range(s)$$
$$range = high - low$$

مثال:

SWISS MISS

Char	Freq	Probability	Range
Space	1	0.1	[0.0,0.1[
M	1	0.1	[0.1,0.2[
I	2	0.2	[0.2,0.4[
W	1	0.1	[0.4,0.5[
S	5	0.5	[0.5,1.0[

Encoder Algorithm

```
low  = 0.0;
high = 1;
while ( ( c = getc( input ) ) != EOF ) {
    range = high - low;
    high = low + range * high_range( c );
    low =  low + range * low_range( c );
}
output ( low );
```

SWISS MISS

C	Low_range(c)	High_range(c)	range	l	h
				0.0	1
S	0.5	1	1	0.5	1
W	0.4	0.5	0.5	0.7	0.75
I	0.2	0.4	0.05	0.71	0.72
S	0.5	1	0.01	0.715	0.72
S	0.5	1	0.005	0.7175	0.72
	0	0.1	0.0025	0.7175	0.71775
M	0.1	0.2	0.00025	0.717525	0.71755
I	0.2	0.4	2.5E-05	0.71753	0.717535
S	0.5	1	5E-06	0.7175325	0.717535
S	0.5	1	2.5E-06	0.717534	0.717535

لفك الضغط:

نأخذ الرقم العشري ونبحث عنه في $range$ المحارف ثم نحسب الـ $number$ سيدلني على المحرف التالي:

$$number = \frac{(number - low\_range(Symbol)}{rage}$$

هكذا في النهاية يصبح $number = 0$.

Dictionary Techniques

طول متغير من $strings$ يكون الـ $token$

أنواعه:

Static Dictionary:

البناء قبل الضغط (لا يتغير).

يمكن التحويل حتى تتناسب مع البيانات التي نري ضغطها.

Adaptive Dictionary (LZW&LZ77):

تبدأ بدون $dicitonary$ أو واحد افتراضي أساسي.

خوارزمية الضغط تضيف عبارات جديدة لاستخدامها لـ $token$ مشفرة.

lZ77

استخدام $window$ وبفر من أجل النظر خطوة للأمام $Sliding\ window\ \&\ look\ ahead\ buffer$

نقارن الـ $window$ ← البفر فيكون لدينا حالتين:

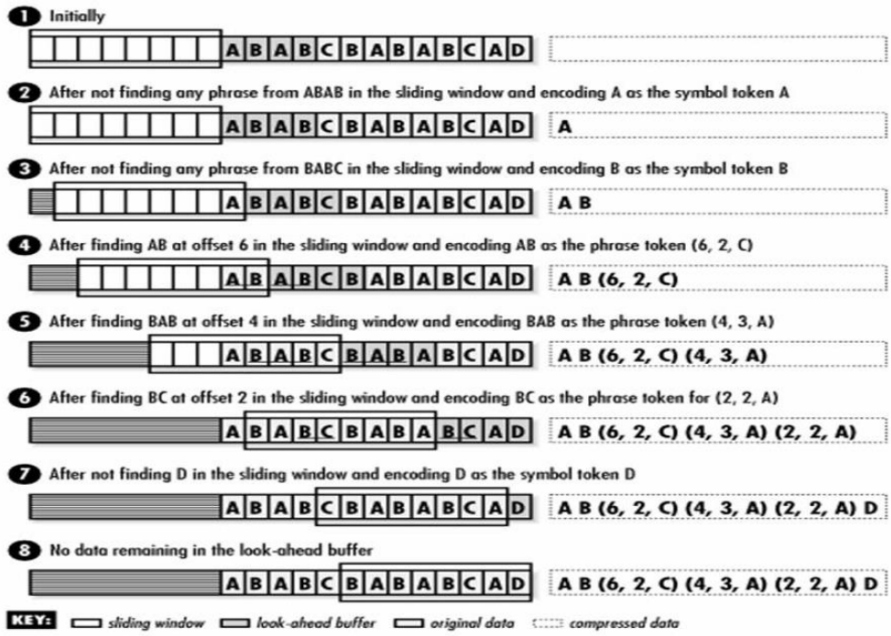
- لا يوجد $Match$: وبالتالي لايوجد ضغط نضع الحرف كما هو.
- يوجد $Match$: نضغط 3 أجزاء من المعلومات:

1. $index$: المحرف الذي بدأ التطابق عنده.
2. طول المحارف التي فيها $Match$.
3. أول محرف مختلف.

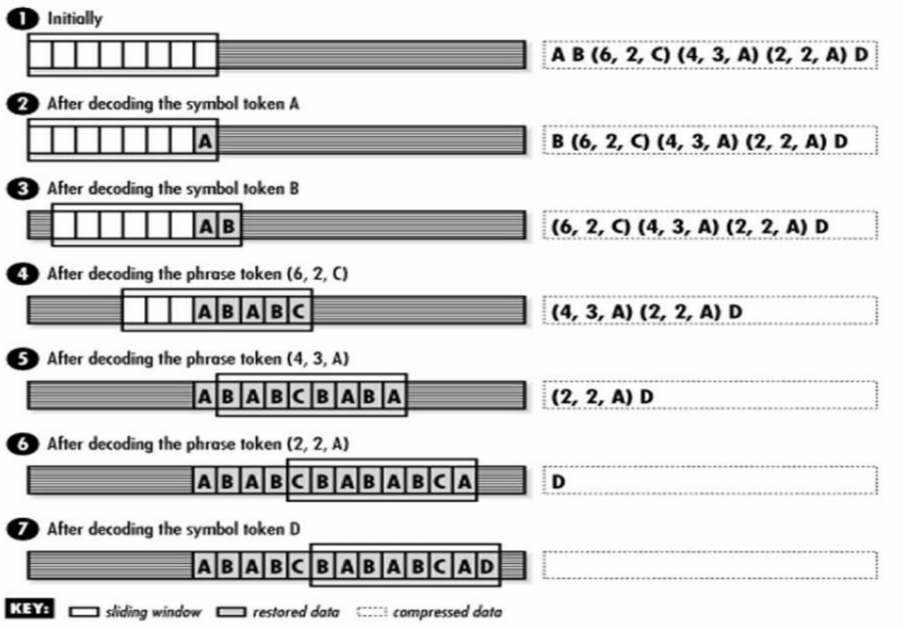
$$(offset, length, symbol)$$

مثال:

Encode LZ77



Decode LZ77



معدل الضغط يعتمد على:

حجم الـ $window$، حجم الـ $buffer$، الـ $entropy$ الداتا،

حجم الـ $window$ عادةً $4KB$.

بفر الـ $look\ head$ أقل من $100B$.

$LZ77$ أفضل من Huffman ولكنها أبطأ (بالفك أسرع)

LZW

لديها قاموس بدائي ونعدل عليه حيث:

طالما وجدت المحرف في القاموس نستمر عدا ذلك نضيف المحرف الجديد مع الذي سبقه كـ $token$ جديد إلى القاموس.

$char\ input\ x$	$code\ output$	$new\ code$
W	"_"	$256 = \_W$
E	W	$257 = WE$
D	E	$258 = ED$
"_"	D	$259 = D\_$
▯	256	$260 = "\_WE"$

حيث:

input: هو المحرف الحالي.

output: هو ما سبقه إذا كان في القاموس أصلا نشكل token جديدة من output + input ونضعه في القاموس

بشرط output +input ليسوا موجودين مسبقا في القاموس ,إذا وجدوا نعود خطوة للسابق.

Decode

يصل رقم نقابله بقيمته في القاموس فينتج الـ *output*.

يبدأ الـ Decodeمع القاموس الذي يحتوي على جميع الرموز (0 إلى 255)، ثم يقرأ المدخلات ويستخدمها لاسترداد عنصر القاموس الأول .

Input Codes :" WED<256>E<260><261><257>B<260>T "

NEW CODE	Output	New Table Entry
' '	" "	
'W'	"W"	256 = " W"
'E'	"E"	257 = "WE"
'D'	"D"	258 = "ED"
256	" W"	259 = "D "
'E'	"E"	260 = " WE"
260	" WE"	261 = "E "
261	"E "	262 = " WEE"
257	"WE"	263 = "E W"
'B'	"B"	264 = "WEB"
260	" WE"	265 = "B "
'T'	"T"	266 = " WET"

Last Output

+

Current Output

=

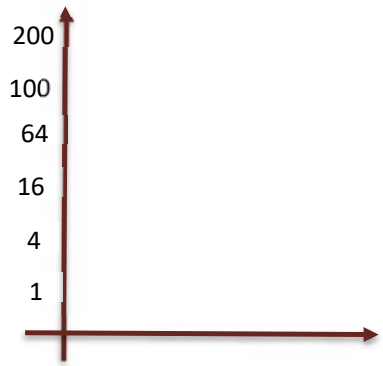
Put in next New Table Entry

Audio Compression

عند ضغط الصورة يهمنا أن تكون الصورة واضحة بدون أن أهتم إذا كانت lossless.

يمكن ان نطبق lossless على الصوت ولكنها ليست فعالة دائماً إلا في حال احتوى الصوت على صمت silence تكون RLE فعالة هنا، أما Dictionary فهي ليست مناسبة أبداً للصوت.

Non-Linear Quantization



في حال التقطيع بشكل linear لا نحتاج لتخزين قيم x نخزن فقط y لأن المسافة بين x تكون ثابتة ولكن في حالة طبقنا ذلك على الصوت فهذا سيعطي الاشارات الضعيفة والقوية نفس الأهمية ولكن أذن الانسان حساسة للإشارات الضعيفة أكثر (حساسة لتغيراتها بشكل أكبر) بينما في القوية نحتاج لتغييرها بشكل كبير حتى نشعر بالتغيير وبالتالي من المهم تقطيع الصوت المنخفض على قيم أكثر.

المنحني y يصبح log.

القيم المنخفضة تأخذ قيم منخفضة على y.

القيم المرتفعة تأخذ فيم مرتفعة على y.

فيكون توزيع النقط على المحور y بشكل غير متجانس قيم كثرة في الأسفل قليلة في الأعلى.

ثم نأخذ القيم نقطعها ونخزنها.

نحن لا نستطيع تحديد قيم الـ y يدوياً لذلك نعدل على الإشارة الأصلية ثم نقطع القيم الجديدة بطريقة خطية.

علاقة إيجاد القيم الجديدة:

- $\mu - law$

$$r = \frac{sgn(s)}{\ln(1+\mu)} \ln\left(1 + \mu \left|\frac{s}{S_p}\right|\right), \qquad \left|\frac{s}{S_p}\right| \leq 1$$

حيث:

$sgn(s)$: الإشارة، $\ln(1+\mu)$: رقم نحن نحدده كلما كان أكبر كان الشكل مقعر أكثر (القيم أكثر).

s: قيمة الاشارة، S_p: أعلى قيمة بالإشارة.

r: هو رقم جديد نقطعه بطريقة خطية.

- $A - law$

$$r = \begin{cases} \dfrac{A}{1 + In\ A}\left(\dfrac{s}{s_p}\right), \left|\dfrac{s}{S_p}\right| \leq 1 \\ \dfrac{sgn(s)}{1 + \ln A}\left[1 + \ln A \left|\dfrac{s}{S_p}\right|\right], \dfrac{1}{A} \leq \left|\dfrac{s}{S_p}\right| \leq 1 \end{cases}$$

حيث: $sgn(s) = \begin{cases} 1\ if\ s > 0 \\ -1\ otherwise \end{cases}$

ملاحظة: في المرجع يوجد اختلاف في A_law الشرط الأول بدل $1 \leq$ الشرط $\frac{1}{A} \leq$ وجب التنويه

لنفترض أن أذن الانسان تلتقط الأصوات من 1 إلى 100,000 بالتالي:

- ✓ تكميم خطي: سنحصل على 100,000 عينة (نحتاج 17 بت).
- ✓ تكميم غير خطي: القفزة بمقدار 10^n: $10^0, 10^1, 10^2, 10^3, 10^3, 10^4, 10^5$

سيكون أقل والتخزين على بتات أقل، فاذا أخذنا 10 عينات بين كل قفزة وقفزة سينتج لدينا 50 عينة تخزن على 6 بت.

Modulation

عندما نريد إرسال إشارة حاملة لرسالة – Message signal، فإننا نرغب ببث هذه الرسالة عبر مسافات بعيدة، وأن توفر قناة اتصال يمكن الاعتماد عليها، وذلك كيلا يحدث ضياع بالمعلومات.

تنتقل الإشارات ذات التردد العالي لمسافات أبعد من الـ Message Signal، وذلك دون أن تتعرض لاضطرابات.

نرغب بتوظيف هذه الخاصية التي تملكها الإشارات عالية التردد، حيث بث الرسالة بالاستعانة بها، وبذلك نسمي هذه الإشارة بالإشارة الحاملة – carrier signal.

تسمى هذه العملية بالـ Modulation.

Pulse-Amplitude Modulation

اسم آخر لعملية الـ sampling،

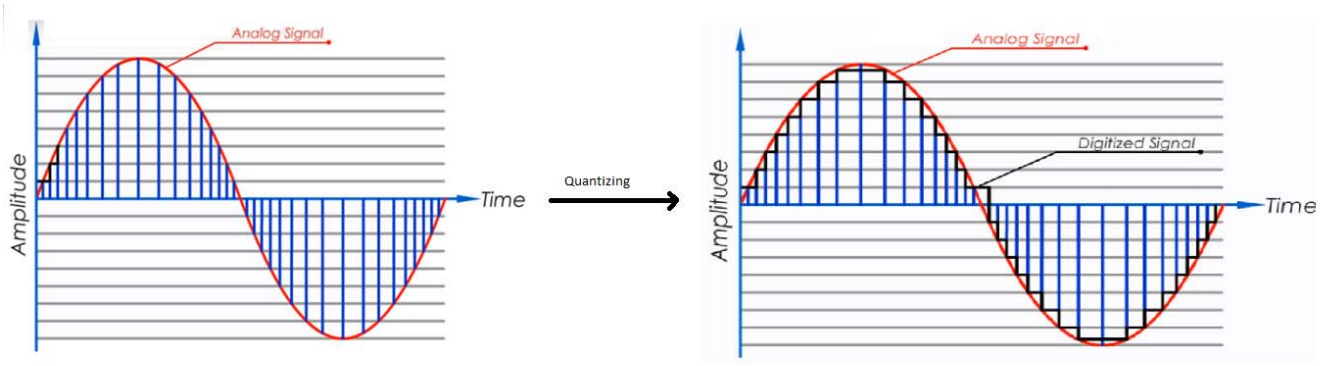
حيث يتم التقطيع وفق سعة الإشارة – Amplitude.

PCM: Pulse-code Modulation

طريقة لتحويل الإشارة التماثلية – Analog Signal إلى إشارة رقمية – Digital Signal، وهي الطريقة القياسية المستخدمة لتمثيل الصوت ضمن الحواسيب. وتتألف من ثلاث خطوات:

1) Sampling.
2) Quantizing.
3) Encoding.

Quantizing - تثبيت القيم



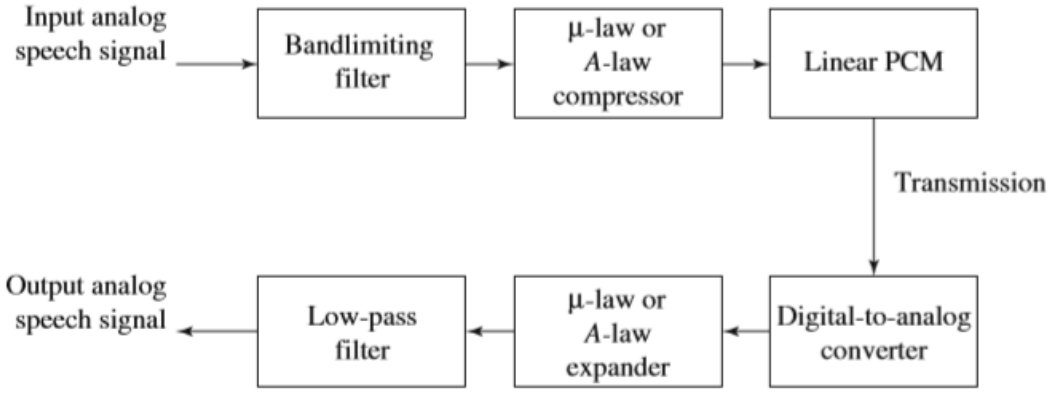
عند التحويل من analog إلى digital، نحتاج إلى تقريب مجموعة من القيم القريبة إلى قيمة موحدة.

Encoding -الترميز

يتم من خلالها تعيين رمز ثنائي – Binary Code لكل قيمة من القيمة الناتجة من عملية الـ Quantizing.

PCM in Speech Compression

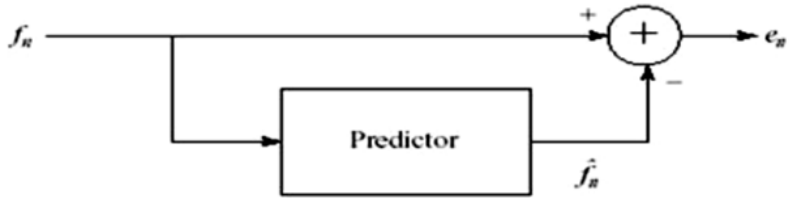
تقطيع الصوت sampling ثم تحويله.



DPCM: Differential PCM

تعتبر أن الفروقات مهمة لأن الصوت لن يرتفع فجأة ولن ينخفض فجأة، وبذلك، ومن أجل التقطيعات المترابطة بشكل كبير، نقوم بإسقاط المعلومات المهملة. يتم أخذ قيمة متوقعة، مستنتجة من الخرج السابق، ليتم دمجها مع القيم المثبتة – Quantized Values.

DPCM Transmitter



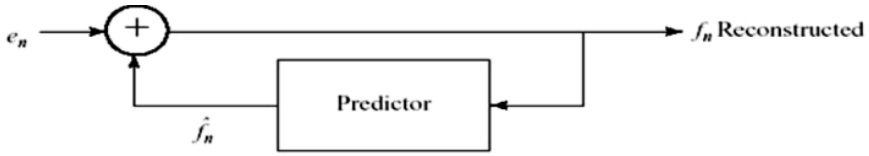
حيث:

f_n: الدخل المقطع – Sampled input.

$\hat{f}_n$: التقطيع المتوقع - predicted sample.

e_n: الفرق بين التقطيع المدخل وخرج الـ predictor، يسمى عادة بـ prediction error.

DPCM Receiver



Delta Modulation

نسخة مبسطة عن الـ DPCM، تستخدم عادة كمحول سريع بين الـ Digital والـ Analog، ويمكن استخدامها فقط إذا كانت التغييرات صغيرة وليست سريعة ونرسل فقط 1 (عند الارتفاع) أو -1 (عند الانخفاض) اي 1bit.

Uniform-Delta Modulation

يتم تطبيق القوانين التالية:

$$\hat{f}_n = \tilde{f}_{n-1}$$

$$e_n = f_n - \hat{f}_n = \hat{f}_n - \tilde{f}_{n-1}$$

$$\tilde{e}_n \begin{cases} +k \text{ if } e_n > 0, & \text{where } k \text{ is a constant} \\ -k \text{ otherwise} \end{cases}$$

$$\tilde{f}_n = \hat{f} + \tilde{e}_n$$

مثال:

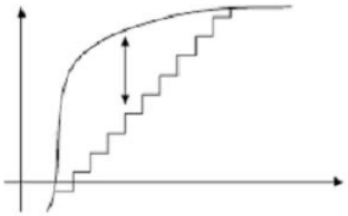
من أجل أربع خطوات (k=4)، وبفرض:

$$\hat{f}_1 = 10, \hat{f}_2 = 11, \hat{f}_3 = 13, \hat{f}_4 = 15$$

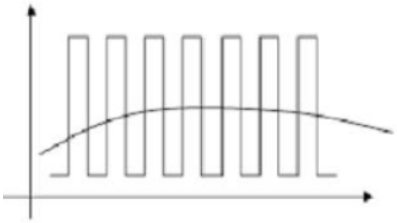
n	$\hat{f}_n$	e_n	$\tilde{e}_n$	$\tilde{f}_n$
1	10	–	–	10
2	11	$11-10=1$	4	$10+4=14$
3	13	$13-14=-1$	-4	$14-4=10$
4	15	$15-10=5$	4	$10+4=14$

من خصائص الـ Uniform-Delta Modulation:

يصعب تعقّب الإشارة من أجل عدد خطوات صغير، أو بسبب التغير سريع للإشارة.



نحصل على تقريب سيء للإشارة من أجل عدد خطوات كبير – k، أو بسبب التغير البطي للإشارة.



ADPCM

يبقى الارسال على 1 بت أي (1/-1) ولكن سيتم الاتفاق بين المرسل والمستقبل على الـ Step.

فإذا وجد خطأ نعود للـ Step الأصلية وإلا نضاعف الـ Step.

عندما يكون الدرج أسفل المنحني الحقيقي للإشارة نضيف الـ step.

- اذا الناتج أقل من المنحني نضاعف حجم الخطوة +1.
- اذا الناتج أكبر من المنحني نعود للـ Step 1-.

الهدف منها ملاحقة الاشارة دون ارسال معلومات اضافية على 1bit (1/-1).

$$+1, +2, +4, +8, -1, -2, -4, +1, +2, -1$$

2x

step عدنا للأصلية

مثال:

An example: (Standard stepsize = 1)

Sender:	7	11	19	21	19	17	14	14	15	14	← actual "curve"
Start value:	5	6	8	12	20	19	17	13	14	16	← $s(t_i)$
Stepsize:		+1	+2	+4	+8	-1	-2	-4	+1	+2	-1
Sender transmits:		+1	+1	+1	+1	-1	-1	-1	+1	+1	-1
Receiver receives:		+1	+1	+1	+1	-1	-1	-1	+1	+1	**-1**

Serious mistakes if transmission error occurs: thus, full information is additionally required from time to time.

Receiver calculates stepsize from run length:

+1 +2 +4 +8 -1 -2 -4 +1 +2 -1

If the receiver has gotten the start value (5 in this example) he may reconstruct the staircase function:

5 6 8 12 20 19 17 13 14 16 .....

القيم في اول سطر هي قيم المنحني ونحصل على value من طرح اول رقمين فينتج 5 , 5 هي اصغر من قيمة المنحني الفعلي لذلك نضيف لها 1 وتصبح القيمة التالية 6 نلاحظ ان 6 اقل من قيمة المنحني الفعلي لذا سنقوم بمضاعفة الخطوة فتصبح 2 ونضيفها للقيمة لتصبح 8 ونقارن مجددا مع 19 .....

عندما أصبحت القيمة 20 وقيمة المنحني الفعلي 19 نقوم بطرح 1 من الخطوة وتصبح القيمة 19 , ولكنها ما تزال اكبر من المنحني الفعلي لذا نقوم بمضاعفة الخطوة وتصبح -2 ثم نضيفها الى 19 فتصبح 17 , نلاحظ ما تزال القيمة اكبر من المنحني الفعلي لذا نقوم بتكرار مضاعفة الخطوة والاضافة وهكذا ...

يتم ارسال قيمة +1 او -1 للخطوة وليس قيمتها والمستقبل يستقبل ذات الشيئ اما قيم الدرج فهي التي قمنا بحسابها.

Delta Modulation vs ADPCM

الخطوة في DM ثابتة وكذلك تقوم بتشويه المنحني ويصبح الصوت مشوشا.

MPEG

نأخذ اذن الانسان بعين الاعتبار، وهي آلية لتنظيم ضغط الصوت مع الصورة معاً (الفيديو).

هناك 4 معايير لأذن الانسان تؤخذ بعين الاعتبار:

- Threshold of audibility عتبة الهدوء
- Frequency masking.
- Critical bands.
- Temporal masking.

مجال سمع الانسان $20\ Hz \rightarrow 20\ KHz$.

أعلى حساسية $1\ KHz \rightarrow 5\ KHz$.

صوت الانسان $500\ Hz \rightarrow 2KHz$.

1. عتبة الهدوء

متى نقول ان هناك صوت وكم يجب أن تكون شدة الصوت لأسمعه هذا متعلق بالـ frequency.

تفيد العتبة بمعرفة الاشارات التي لا نحتاج لتخزينها.

منحني العتبة يأخذ frequency وdBوبالتالي نحن بحاجة للـ freq.

2. Frequency Masking

Mask width: $100\ Hz \rightarrow 4KHz$

كل freq له انطباع (أي بظل هذا التردد لن أسمع شيء).

كل freq له rang ليقوم بـ Mask عليه، أي:

Freq بهذه الشدة وهذا المقدار يقوم بـ Mask.

هناك freq عندما تأتي مع الـ freq mask لا نسمعها بالتالي لا حاجة لتخزينها.

Band هو المجال الذي يقوم الـ freq بعمل mask له.

3. Temporal Mask

انطباع متعلق بالزمن (كلما زادت الشدة أو زمن استمرار الصوت يزداد الانطباع).

كلما نسمع صوت ويتوقف فإنه سيأخذ فترة حتى يزول (انطباع)

كل الاشارات الأقل من الأصلية وجاءت بعدها بالزمن لن نسمعها.

$1KHz\ at\ 60\ dB \rightarrow 1.1\ KHz\ at\ 40\ dB \rightarrow$ لن نسمعها

Moving Picture expert group (MPEG-1)

تتألف من 3 طبقات تشفير مختلفة |,||,||| layer كل **layer** تضم السابقة ولكل واحد خوارزمية ضغط مختلفة في الفك.

الطبقة N تفك كل الـ layers التي قبلها كما أن التعقيد يزداد من طبقة إلى أخرى.

Layer1: أقل تعقيداً، بسيطة، لا تأخذ الأذن بعين الاعتبار، سريعة لا يهمها الحجم بل السرعة مع الحفاظ على جودة الصوت.

Layer2: أكثر تعقيداً، نأخذ الأذن بعين الاعتبار، تستخدم لبث الأصوات (broadcast) على الانترنت، نحتاج حجم قليل في البث.

Layer3: تعقيدها أعلى، تطبيق **Huffman** لتكميم القيم.

نسبة الضغط 6:1

فك الضغط Real Time لكن من الصعب ان يكون الضغط Real Time.

ان معيار MPEG يوصّف فك الضغط مع توضيح خطوات الضغط وهو موحد مهما كانت طريقة الضغط (هاردوير).

MPEG-1 ليست مخصصة لإشارات الكلام (صوت) الهدف هو استرجاع الاشارة بجودة جيدة وتكون مشابهة للاشارة الأصلية وليست نفسها.

الخطوات:

1. DFT (512 samples)
2. نأخذ Sub bands (32)
3. نعرف Mask
4. لكل sub band نعرف رقم يشير إلى شدة الصوت في مجال freq هذا الـ sub band.
5. هذه الأرقام تكون مكممة.

MP3

- MP3 is MPEG-1 in layer|||.

- ضغط ممتاز وفك أسرع وأبسط.
- مناسبة للـ CD.
- الـ Decoder فقط هو الموحد Standardized.

MP3 Stereo

أبسط حل كل اشارة ترمز لوحدها، أو نرمز متوسط الاشارتين ثم فروقات الاشارات عن المتوسط او فرق أحدهما عن الأخرى.

أو freq المنخفضة تبقى نفسها أما العالية نرمز متوسط وفروقات.

MP3 جودة عالية Band Width عالية.

الفيديو

الفيديو: هو عبارة عن صوت + سلسلة من الصور، وعادة ما يكون الصوت وضغطه منفصلين عن الصورة وضغطها.

عندما نقوم بتقطيع الفيديو من أجل تحويله لسلسلة صور رقمية فـ نحتاج للتقطيع على بعدين:

1. **بعد مكاني:** تقطيع على إحداثيات الصورة (x,y).
2. **بعد زماني:** أي frames ويقصد هذه الصورة بهذه اللحظة.

وعند التقطيع يحدث ضياع بالمعلومات بين الصورة والأخرى.

- الفيديو عملياً عبارة عن مصفوفة ثلاثية البعد من الـ pixels $(x.y.t)$:
  - ✓ بعدين مكانيين $(width\ ,height)$
  - ✓ بعد زماني (across frames) أي يحدد بأي frame جاء هذا ال pixel.

وهذه القيم هي عبارة عن ألوان.

عند تشغيل الفيديو تترك الصورة الأولى انطباعا في العين قبل عرض الصورة التالية فكم **frame** نحتاج لنرى حركة الصور بحالة مستمرة؟

الجواب: نحتاج إلى 15 frame في الثانية حتى لا تشعر العين بالتقطيع أي دون هذا الحد سيكون التقطيع واضحاً بين الصور.

- ✓ عند 24 fps (frame per second) يكون هناك اهتزاز jerkiness، خاصة بالأجسام التي تتحرك بسرعة وقريبة من العين.
- ✓ عند 30 FPS الحركة تكون انسيابية smooth.
- ✓ حالة مثالية عند الـ 50fps.

نستنتج مما سبق أن كلما زاد عدد الـ frames كلما زادت انسيابية الحركة.

Refresh Rate:

الصورة بحاجة أن يتم القيام بـ refresh لها بشكل مستمر على الشاشة وبمعدل refresh عالي وذلك من أجل خداع العين (متعلق بالهاردوير).

آلية الرسم على الشاشة

- ➢ نرسم على ذاكرة ونعرض على ذاكرة أخرى وذلك للتخلص من رفة الشاشة (flickering).
- ➢ الصورة في كل مرة ستمحى وترسم صورة جديدة فكلما حدثنا الفورم يتلون كله بلون الخلفية ثم ترسم الصورة الجديدة وهكذا..
- ➢ نتيجة لذلك العين سترى مرة الصورة مرة اللون(الرفة).
- ➢ المشكلة هنا أننا نمسح أمام العين فالحل أن نمسح على مسودة (ذاكرة خلفية).

إذاً لدينا ذاكرتين:

1. ذاكرة فيها كل بكسل لون.
2. ذاكرة فيها كل بكسل ماذا سيصبح لونه.

تم swap بين الذاكرتين (نقلب عنواني الذاكرتين) والـ swap غير مكلف ويحدث بلحظة.

أول scan الصورة الأولى وثاني scan الصورة الثانية وبالتالي لم نرى الانتقال سنرى فقط الصورة وتغيرها.

Computer: uses refresh buffer at 70 Hz

TV: uses 50 Hz or 60 Hz

مثلاً التقطيع على 15 frame ولكن الـ refresh = 60Hz أي سنقوم بـ refresh للذاكرة التي تحوي الصورة 60 مرة.

Aspect Ratio

$$Aspect\ Ratio = w/h$$

أي نسبة عرض الصورة على طولها.

- ✓ معدل نسبة الطول إلى العرض التقليدي المستخدم في التلفاز هو 4:3
- ✓ أما الأنظمة الحديثة تستخدم معدل 16:9

Video Signals

طرق نقل الإشارة:

1. Component video:

- ✓ لدينا ثلاثة إشارات (مآخذ) فيديوية منفصلة (R,G,B)
- ✓ نحتاج إلى ثلاثة وصلات وموصلات لتوصيل الكاميرا إلى المراقب.
- ✓ نحتاج إلى أكبر عرض حزمة (Bandwidth) ونحتاج أيضاً إلى تزامن جيد من المكونات الثلاثة حيث نحن بحاجة إلى استلام إشارات مهمة (R,G,B).

2. Composite Video:

- ✓ إشارتي اللون (Chrominance) والكثافة (luminance)
- ✓ (Chrominance) مركبة من مكونان لونيان اثنين (I and Q , or U and V)
- ✓ في موصل ال RCA الأبيض والأحمر موصولان للاستخدام في مسجلة الصوت (stereo audio)

أي يكون لدينا مأخذين للصوت stereo.

- ✓ على سلك واحد سنرسل أكثر من إشارة بالتالي نعطي لكل إشارة frequency معين.
- ✓ اليوم على نفس الـ frequency نستطيع تحميل أكثر من إشارة وأكثر من frame مثلا الـ 300 حيث كل قناة تستقبل 30 (وهذا هو سبب وجود أكثر من قناة تلفزيونية على نفس الـتردد).

3. :S-Video

- ✓ فيها سلكين، أحدهما للون chrominance والآخر للإضاءة luminance.
- ✓ السبب في ارسال الإضاءة y في سلك لوحدها أن العين حساسة للضوء أما بيانات الألوان فنستطيع ضغطهما وحتى الحذف منهما (u,v).
- ✓ نرسل y لكل البكسلات أما الـ v, u فنرسل لبكسل القيمة u وللبكسل الآخر القيمة v.

أثناء العمل يستطيع كل بكسل الاستعارة من جاره قيمة v or u وبالتالي سنبعث v, u أقل من المطلوب.

آلية الإرسال والاستقبال

1. نأخذ الصورة RGB.
2. نحولها لنظام لوني آخر مثلاً: yc_1c_2.
3. تصبح الإشارة إضاءة ولون.
4. Multiplexing دمج (لون + إضاءة + صورة).
5. نبثها.
6. استقبال.
7. فك.
8. تحليل ثم عرض.

Analog Video

آلية عرض للفيديو بطريقة analog، الاعتماد هنا على شاشات الـ CRT (شاشة خلفها مدفع)

داخل هذه الشاشة يوجد مكثفات، ثم بإطلاق شعاع باتجاه الشاشة تقوم المكثفات بتحريكه حتى يمسح الشاشة كاملة كلما تواجد الشعاع بنقطة يلونها.

مسؤولية هذا الشعاع أن يعود للنقطة الأولى قبل زوال اللون (أي يستفيد من زمن الانطباع).

3 أشعة (شعاع لكل لون).

أسلوب المسح هنا يسمى (المسح النقطي)، تنقسم الشاشة إلى خطوط أفقية تنبعث ثلاثة أشعة إلكترونية للألوان الثلاثة الأساسية, خطوط المسح ليست أفقية لأنه يتم استخدام جهد كهربائي صغير، مما يحرك شعاع الإلكترون لأسفل بمرور الوقت

أنظمة البث الخاصة بـ Analog لبث الألوان والأشعة:

1. NTSC:
   - 525 خط مسح (Scan line)
   - عدد الخطوط بالـ frame الواحدة 30 fps
   - تستخدم نظام الألوان YIQ
2. PAI:
   - 625 خط مسح (Scan line)
   - عدد الخطوط بالـ frame الواحدة 25 fps
   - تستخدم نظام الألوان YUV (U&V على نفس الحامل)

3. SECAM:
   - 625 خط مسح
   - عدد الخطوط بالـ Frame الواحدة 25 fps
   - تستخدم نظام الألوان YUV (U&V كل منهم على حامل مختلف عن الآخر)

ظهرت مشاكل أن الهاردوير لم يستطع التعامل مع حجم الـ Frame الكبير لذلك أصبح الرسم على الخطوط الفردية أولاً ثم الزوجية (نظام مسح متشابك interlaced)

ولكن هذا تسبب بمشاكل للأجسام المتحركة

الحل هو Progressive Scanning:

الخطوط lines تكتب في frame buffer بشكل تدريجي

لم تعد مشكلة الـ flickering موجودة وبالتالي يمكن استخدام frame rate منخفضة.

شاشات plasma وLCD تستخدم هذه الآلية.

CRT ممكن أن تستخدم Progressive أو interlaced.

Analog vs Digital

Analog: يستخدم الأمواج المغناطيسية.

Digital: المعلومة ترسل على شكل bit كل نقطة فيها معلوماتها.

Digitizing

للتحويل من Analog إلى Digital:

1. يصل الفيديو Analog
2. نأخذ الـframe
3. نقطعها
4. نحولها لـ Digital
5. نقطع Digitized frame
6. نضغط

ميزات Digital Video

- ✓ دقة عالية للصور وجودة عالية للصوت.
- ✓ قابلية تخزينها على أجهزة ديجتال.
- ✓ إمكانية الضغط وفك الضغط بسهولة.
- ✓ متوافقة مع أجهزة الكمبيوتر والإنترنت.
- ✓ لكنها تتطلب هاردوير وسوفت وير وسعة تخزين عالية.

Chroma Subsampling

- العين أكثر حساسية للضوء من اللون لذلك نقطع الألوان (U,V) أكثر من الإضاءة Y.

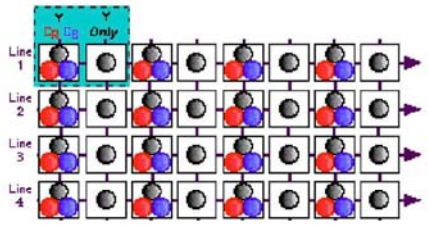
عادة (4:2:2) يعني لكل 2 بكسل أرسل YUV كاملة (تقطيع افقي)

المستقبل ممكن أن يكرر قيم U وV من مجاوريهما.

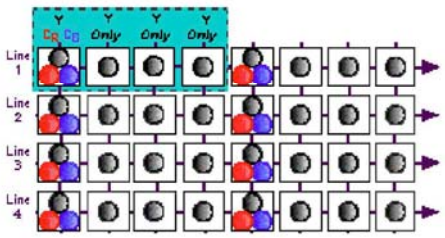
(4:1:1) لكل 4 بكسلات أرسل YUV وهو تقطيع افقي

(4:2:0) تقطيع طول وعرض سوياً (مرة سنرسل U ومرة سنرسل V)

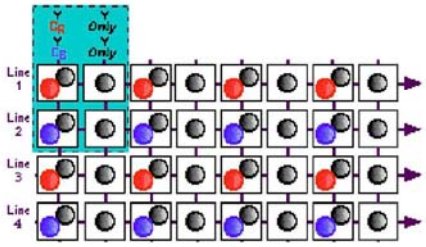
- 4:4:4 – No subsampling
- 4:2:2, 4:1:1 – horizontally subsample
- 4:2:0 – horizontally and vertically



4:2:2



4:1:1



4:2:0

HD TV

برامجها يمكن أن تتضمن Dolby ديجتال المستخدمة في DVD والسينما.

لها Bandwidth أكبر بـ5 الى 8 مرات من NTSC/PAI

Codec Compressor/Decompressor

خوارزمية تقلل من حجم الفيديو لتسمح بعرضه على الحاسوب أو الانترنت.

معظم الـ Codec تستخدم الـ Lossy Compression Methods.

Video Compression

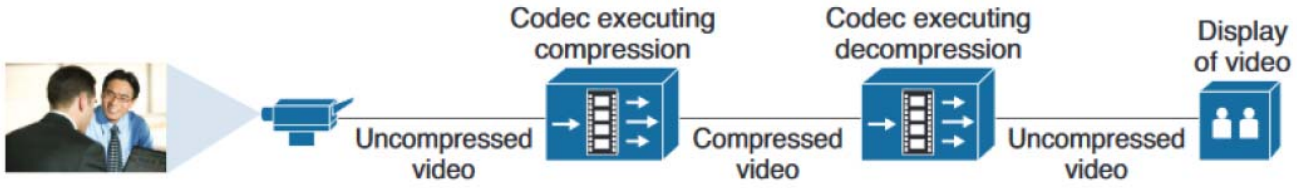
فائدة الضغط:

لبث فيديو HD سوف نحتاج إلى:

$$\left(\frac{720 \times 1280\ pixels}{frames}\right) \times \left(\frac{60\ frame}{sec}\right) \times \left(\frac{3\ colors}{pixel}\right) \times \left(\frac{8\ bits}{color}\right) = 1.3\ Gb/s$$

نحتاج إلى 1.3 Gb/s ولا يوجد صوت بعد!

عرض القناة هو 20 Mbps وبالتالي نحتاج لمعدل ضغط 70:1



عندما نضغط نقوم بحذف التكرار، والتكرار في الفيديو على بعدين:

Spatial Compression & Temporal Compression

1 – المكاني Spatial Compression:

نحذف الألوان لأنها هي الوحيدة التي تتكرر وهذا بالتحديد ما تقوم به صيغة JPG

JPEG يمكن بشكل مثالي أن تحقق تخفيض بنسبة 90 ٪ أو 95٪ في حجم ملف الصورة دون خسارة مرئية في الصورة.

2 – الزمني Temporal Compression:

تعتمد على الفرق بين الـ Frames (أي إذا وجد 2 frames لم يتغيرا عن بعضهما لا نقوم بإرسال الـ 2).

Compression - الضغط

يمكن التنبؤ بالبكسلات الجديدة وحساب الفرق بين الصورة التي قمنا بالتنبؤ بها والصورة الأصلية ونرسل هذه الفروقات (عدد بتات أقل).

Temporal Prediction:

يمكن تقسيم الـ frames إلى blocks ونرسل تحركات هذه البلوكات (أشعة Motion).

حيث أن الـ block ممكن أن يكون بموقع في الـ frame الحالي وينتقل لموقع آخر في الـ frame التالية.

أيضاً من الممكن أن نتوقع كل frame من الـ frame السابق ونقوم بحساب الفروقات (الأخطاء).

أخطاء الفروقات من الممكن أن نعتبرها صورة ونضغطها بصيغة JPEG مثلاً ويتم ترميزها باستخدام بتات أقل.

المناطق التي لم يتم توقعها بشكل جيد ترسل كما هي.

في حال وجد الـ Matching نرسل الحركة، وإذا لم يوجد نرسل الـ block كما هو.

Key Idea

Intra frame (I Frame):

نعامل كل frame بشكل مستقل (لا علاقة لها بالضغط الزماني).

نضغطها بشكل عادي كصورة ونرسله.

Predictive Frame (P Frame):

الـ frame يتم التنبؤ بها من الـ frame السابقة فهي غير مستقلة (من الممكن أن تكون I Frame أو P Frame).

الـ Macro block الحالي يتم التنبؤ به من macro block مشابه في I أو P سابق والفرق بين ال macro blocks يتم ترميزه.

آلية الإرسال:

أول frame نقوم بإرساله كـ I Frame (Intra frame) كما هو، ثاني frame نرسل الفروقات عما قبله، وثالث frame الفرق عن الفروقات التي قبله.

في لحظة ما سوف نضطر أن نتوقف، ونستقبل Iframe لتعديل الخطأ الحاصل بسبب PFrame.

أيضاً يفيد وجود الـ Iframe في عملية الـ Seeking، عادة يتم استعمال 2 Iframes لكل ثانية وذلك من أجل عملية الـ Seek فعند الانتقال في الفيديو نحتاج للوصول إلى Iframe حتى نستطيع رؤيتها.

Intra Frame Compression

1 – نحول من الـ RGB إلى الـ YUV

2 – نقطع لـ Macro Blocks (16x16 Blocks)

3 – تحويل بـ DCT (Discrete Cosine Transformation)

4 – التكميم Quantization

5 – Entropy Coding

تشبه الـ JPEG.

البيانات هامة ولكن رغم ذلك يوجد خسارة

Predictive frame (P frame)

هنا يتم المقارنة بين الـ frames يوجد احتمال كبير ليتشابه كل frame مع ما يليه.

نأخذ الفروقات بين الاطارات ونضغطها.

إذا كانت الـ frame المتوقعة مختلفة جداً عن الـ frame الحقيقية نقوم بتشفيرها بشكل مستقل نعتبره (1 frame).

الخطوات:

1. **الاختلاف:** نطرح بكسلات ثم نضع عتبة إذا كان الفرق أصغر من العتبة فلا نعتبر أن هناك تغيير، إذا كانت الفروقات عالية نضغط الـ frame الحالي وتفشل إذا وجدت حركة Motion
2. Block Differencing:

نقسم الصورة إلى بلوكات من بكسلات نقارن Block B في الـ frame الحالي مع الـ Block P في الـ frame السابقة (نحتاج آلية Matching N) في حال كانت الفروقات كبيرة سنرسل احداثيات الـ Block التي وجدتها Matching يعني لن نرسل البلوك P بل سنرسل احداثيات أقرب بلوك لها والفروقات (طبعاً هنا الاختلاف كبير).

3. Motion Compensation:

نتوقع حركة البلوك بين الـ frames.

الحركة تمثل عن طريق Motion vector.

يتم ضغط Motion vector عن طريق DCT.

سنرسل الفروقات والـ vector.

حتى تكون فعالة يجب أن تكون الحركة فقط translated ليست rotate أو scale (تشوه لن يكون هناك Match).

تغيُر الإضاءة سيسبب مشكلة لأن الفروقات ستكون عالية جداً وسنضطر لتوليد frame I جديد.

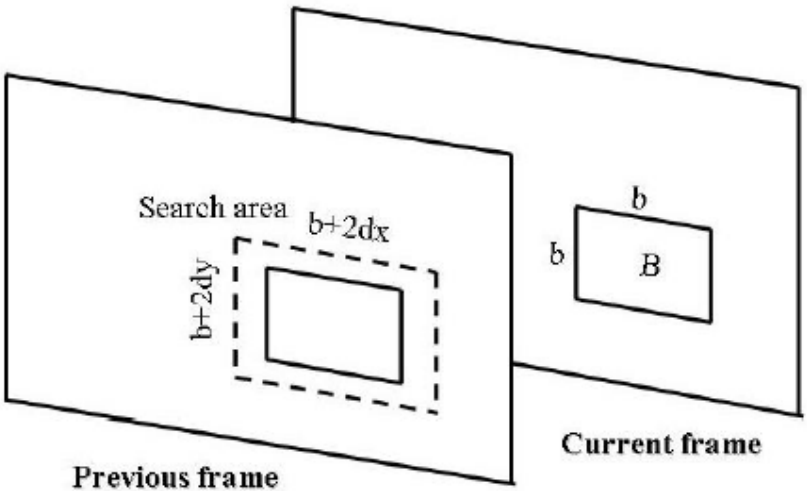
لكل بلوك سنرسل شعاع الحركة والفروقات بين بكسلاته عن بكسلات أقرب بلوك له.

قيم الأشعة هي قيم صغيرة ومترابطة أي المناطق التي حركتها عالية حولها أماكن حركتها عالية، هذا الترابط يساعدني في عملية الضغط.

نحتاج لمعرفة آلية التقطيع.

البلوكات الكبيرة تقلل من احتمال الـ matching والصغيرة تزيد أشعة الحركة (زيادة حجم)

عملية البحث: للبحث عن البلوك B في الـ frame السابق في المنطقة $(b + 2dy, b + 2dx)$



4. Distortion Measure:

من أجل تحديد أفضل بلوك للـ Match نقوم بقياس البعد بين بلوك وآخر، لحساب هذا البعد يوجد معادلتين:

a. MAD (Mean Absolute Difference):

$$\frac{1}{b^2}\sum_{i=1}^{b}\sum_{j=1}^{b}|B_{ij} - C_{ij}|$$

b. MSD (Mean Square Difference):

$$\frac{1}{b^2}\sum_{i=1}^{b}\sum_{j=1}^{b}(B_{ij} - C_{Ij})^2$$

نأخذ البلوك صاحب أقرب بعد نعتبره Match.

يوجد عتبة إذا البعد كان أصغر منها هنا الصورة نفسها وإلا نرسل الفروقات.

آليات البحث

Full Search:

البحث في كل المجال $2dx, 2dy$.

نحرك البلوك 1 بكسل ضمن المجال.

Distance Dilated Search:

نبحث حول البلوك، ونقارن مع بلوكات أقل كلما ابتعدنا عن موضع البلوك الأصلي، إذ أن احتمال المطابقة يتقلص.

Locality-based search:

عندما نصل إلى أفضل match، نعيد تنفيذ الخوارزمية ولكن انطلاقاً من المنطقة المحيطة بالـ match الذي تم إيجاده.

2D log Search:

نحدد أربعة من البلوكات المجاورة، ثم نقوم بإعادة تنفيذ الخوارزمية عودياً انطلاقاً من أقرب match، وذلك مع تقليص حجم الخطوة كل مرة.

Step size

$$s = 2^{\log d - 1}$$

مثال:

من أجل البلوك $B(a, b)$، فإنّنا نحاول المطابقة مع البلوكات:

$$(a, b), (a, b + s), (a, b - s), (a + s, b), (a - s, b)$$

ونقوم بتقليص الخطوة كل مرة، وذلك من خلال: $s = \frac{s}{2}$.

حساب شعاع الحركة – Motion Vector Calculation:

فور الانتهاء من المطابقة – Matching، نحسب البعد بين رأسي البلوكين.

ترميز الشعاع – Coding Motion Vectors:

يجب أن تكون عملية الترميز وفق إحدى الطرق: Huffman, Athematic، أو مثيلاتها.

:Coding Prediction Error

يمكن أن تتم عملية الإرسال وفق DCT، أو أن يتم الإرسال بشكل صريح.

مزايا Motion

1. الـ $Motion$ تخفض الـ $bitrate$.
2. إضافة تعقيد (خوارزمية $Matcing$ مكلفة).
3. تحتاج ذاكرة لتخزين الـ $frame$ التي قبل والتي بعد.

variable bitrate

هنا $frame$ تأخذ $bitrate$ أكثر من غيرها.

MPEG − 1 Video

- تستخدم 4: 0: 2 للـ $Subsampling$.
- تدعم $random\ access\ frame$.
- تدعم عكس الفيديو $playback$.
- تدعم $first\ forword$، $fast\ reverse$.
- تملك $I\ frame$، $P\ frame$، $B\ frame$.

MPEG I frame (معيار يعني في الفيديو الصوت ...)

- ترمز باستخدام المعلومات الموجودة بها فقط.
- تدعم $random\ access\ point$.
- معدل ضغطها متوسط (كل $I\ frame$ تضغط بـ $Jpeg$).

$$I\ frame(RGB) \rightarrow y\ c_b c_r \rightarrow Macro\ block \rightarrow each\ block \rightarrow DCT \rightarrow Quantaization$$
$$\rightarrow Zig\ Zag \rightarrow RLE \rightarrow Huffman$$

$MPEG-1\ P\ frame$ (ترسل فروقات)

- ترمز بالارتباط مع أقرب $frame(I/P)$.
- كل $block$ لها $motion$ 1.
- ضغطهما عالي (بسبب استخدام الـ $Motion$),
- الـ $Stream$ يوجد بهما $I\&P$ وبينهم $B\ frame$.
- أخطاء كثيرة (السبب استخدام $P\ frame$ كل مرة كمرجع).

تزايد أخطاء كل P يعتمد على P

$MPEG-1\ B\ frame$

- نأخذ أقرب $(I/P)frame$ قبله وأقرب $(I/P)frame$ بعده,
- سنرسل فروقات عن $2frame$.
- لن يكون هناك تزايد بالإخطاء متل الـ $P\ frame$.
- ضغط أعظمي.
- توقع على جهتين (حتى الأجسام الغير متحركة وستتحرك مستقبلاً نستطيع القيام بـ $Match$ منها).
- نقارن مع أول $(I/P)frame$ قبل ومع أول (I/P).
- بعد أو مع الـ avg بين السابق والتالي.
- إذا وجدنا $Match$ للسابق والتالي (معاً كلاهما)، نرسل $2\ Motion\ vector$.
- إذا $Match$ لأحدهما، نرسل $1\ Motion\ vector$.

Pattern:

سيل من الـ frames، يكون عدد الـ frames بين الـ Iframe والـ Iframe التالي هو N، ويكون عدد الـ frames بين الـ I/p frame والـ I/p frame التالي هو M.

الفك:

عند إرسال frames، فإنه لن يتم إرسالها بالترتيب، ويترتب على الـ encoder مهمة إعادة الترتيب ليتم عندها فك ترميزها من قبل الـ decoder.

عملية الـ Encoding:

عادة لا تكون العملية real-time، ويمكن أن تكون real-time باستخدام معالجات تفرعية.

عملية الـ Decoding:

تستخدم برامج الترميز.

MPEG-1 Video Stream Structure

يدعى أيضاً بالـ bitstream، ويملك 6 bit، خصائصه:

1. Sequence Layer.
2. Gap Layer تبدأ بـ I/P وتنتهي بـ I/P header، فيه عدد الصور.
3. Picture Layer لها header يحوي أسماء الخوارزميات ومعلوماتها.
4. Slices، تملك header.
5. Macro block، تملك header.
6. Block، تملك header.

H-261

موجهين للحركة القليلة، وعند bandwidth محدود. يستخدم Iframe, Pframe فقط. يتم استخدامه في مكالمات الفيديو.

خصائصه:

- يبدأ بـ Iframe جديد كلّما أصبح الخطأ كبيراً.
- Subsampling: 4:2:0.
- يتم الإرسال بالترتيب، والفك يتم بالترتيب الذي تم الاستقبال وفقه.

نهاية الملحق..