Image compression

د. مدحت الصوص

RB Informatics; 1/6/2022 MultiMedia

عناوين المحاضرة:

- Images Lossless Compression.
- Images Lossy Compression.
- Transform Coding & DCT.
- JPEG Compression.
- JPEG Modes.

- عند ضغط الصور يمكن استخدام خوارزميات ضغط lossless او lossy، وعند استخدامنا لخوارزميات الـ lossy نحاول أن نضغط الصورة بحيث يكون تغيرها بسيط بالنسبة للعين، أي أن العين لا تلاحظ التغيير بعد الضغط بشكل كبير، بدايةً لنتحدث عن خوارزميات ضغط lossless

Images Lossless compression

1. خوارزمية RLE:

يمكن ان تستخدم هذه الخوارزمية لضغط الصور بشكل lossless أو lossy

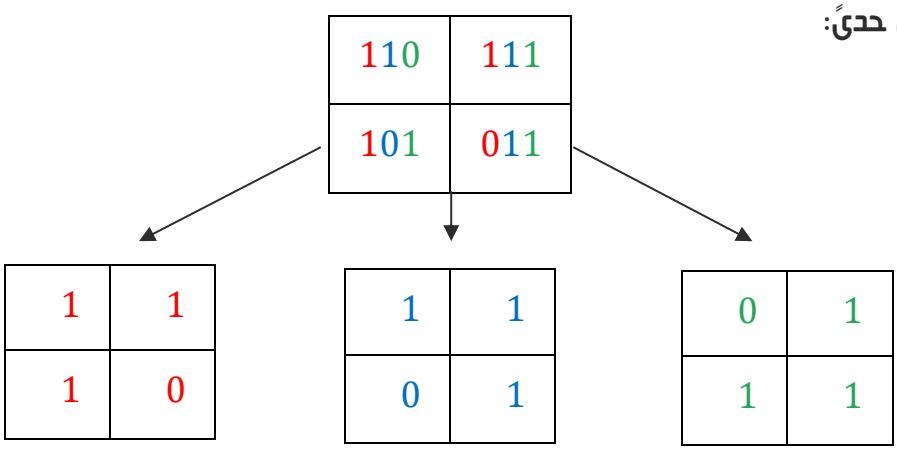
- لضغط صور binary (ابيض واسود) بهذه الخوارزمية:

الصورة تكون 2d ولكن في هذه الخوارزمية يجب أن يكون لدينا مصفوفة أحادية لذلك نقوم بمسح الصورة سطر سطر لنحصل على سلسلة (ويمكن ان نقوم بأخذ أعمدة بدلاً من الأسطر المهم أن نحصل على تتالي أحادي) ثم نقوم بتطبيق الخوارزمية على هذه السلسلة.

- لضغط صور grayscale بهذه الخوارزمية (باستخدام n bit):

على فرض أن كل pixel في الصورة يمثّل ب n bit نقوم بفصل الصورة الاصلية إلى n صورة بحيث نأخذ لكل صورة بت واحد من كل pixel أي نأخذ من pixel الأول أعلى بت ومن الثاني أعلى بت ومن ثم للصورة الثانية نأخذ من كل pixel ثاني أعلى بت وهكذا...
ثم نقوم بضغط كل صورة منهم على حدى.

مثال: لتكن لدينا الصورة grayscale والتي يمثل فيها كل pixelب 3 bit فنفصلها إلى3 صور Binary ثم نقوم بضغط كل منها على حدىً:



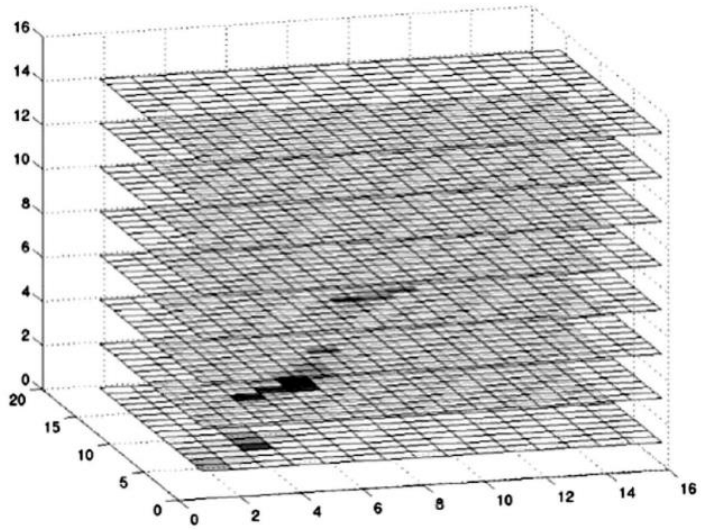
ال bit الثالث من كل pixel ال bit الثاني من كل pixel ال bit الأول من كل pixel

ما الهدف من عملية تقسيم صورة grayscale إلى عدة صور binary ؟

بما أن خوارزمية RLE فعّالة أكثر عند تتالي سلاسل طويلة من نفس الرمز ("0","1") فإن هدف عمليّة تقسيم الصورة هو محاولة تجميع أكبر عدد من هذه سلاسل إلا أنه عندما يكون لدينا بكسلين متجاورين لا يكون الفرق اللوني كبير بينهم (إلّا إذا كان البكسلين على طرفي حافة).

- **مثلاً**: لدينا بكسلين متجاورين ترميز الأول 110 وترميز الثاني 111 نلاحظ أن أول bit (MSB) لا يتغير لأن اللونين قريبين من بعضهما على عكس آخرbit (LSB)والذي يمكن أن يتغير بسهولة لأن تغيره لا يغيّر لون ال pixel بشكل جذري وهكذا أصبح بإمكاننا الاستفادة من هذه الخاصية عند فصل الصورة الاصلية إلى n صورة فتكون غالباً كل صورة تحوي سلاسل متتالية من نفس الرمز.
- لكن على الرغم من صحة الكلام السابق نظرياً، إلا أن هنالك مشكلة عملية في تطبيقها، وهي أنه يمكن أن تختلف البتات كلها لرقمين متجاورين ممثلين في النظام الثنائي مثل 0111,1000 وهذا سوف يسبب مشكلة عند تقسيم الصورة الأصلية، لحل هذه المشكلة نلجأ إلى تمثيل البكسلات ب Gray code حيث في هذا النظام يكون الفرق بين أي رقمين متجاورين هو bit واحد فقط، أي لن نقع في المشكلة السابقة.

- RLE for grayscale images



Decimal Number	4 bit Binary Number ABCD	4 bit Gray Code $G_1G_2G_3G_4$
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

2. Statistical Methods:

نفس الخوارزميات التي طبقناها على النصوص يمكن أن نقوم بتطبيقها أيضاً على الصور:

القيم المتكررة يتم تخزينها باستخدام أقل عدد من pixels.

تعمل هذه الخوارزمية عندما يكون للقيم احتمالات مختلفة: القيم المختلفة لها نفس الاحتمالات تقريبًا.

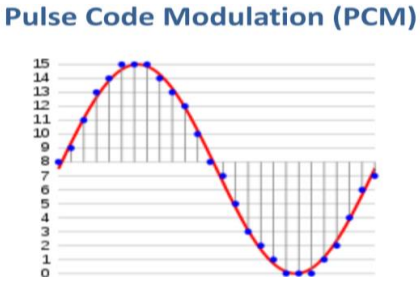
الألوان المتشابهة لها قيم مختلفة.

وهذه الخوارزميات في المجمل ليست خيار جيد لضغط الصور.....

Images Lossy compression

- عند تطبيقنا لهذه الخوارزميات يجب أن نركز على أن تكون الصورة بعد الضغط لا تختلف كثيراً على عين الانسان، فكما أن الاذن حساسة أكثر للترددات العالية من الترددات المنخفضة، كذلك العين تتحسس لتغيرات الاضاءة أكثر من تغيرات الألوان، وبهذا نحاول عند ضغط الصورة أن تكون حساسية العين للتغيرات بعد الضغط أقل ما يمكن حيث ضغط الصور يعتمد على حقيقة ان البكسلات المتجاورة تكون مترابطة بشكل كبير.

1. Pulse Code Modulation (PCM):



الفكرة منه هو تقطيع الإشارة على محور الx ومحور ال y حيث يتم أخذ عينات من سعة الإشارة التناظرية على فترات منتظمة ثم يتم تكميم كل عينة إلى أقرب قيمة من ضمن مجال digital steps.

2. Linear Pulse Code Modulation (LPCM)

هو حالة خاصة من PCM حيث يقوم بالتكميم على مستوى y وتفصل بين المستويات التكميمية مسافات ثابتة.

ملاحظة: على الرغم من أن ال PCM هو أعم من ال LPCM إلا أنه عادة يستخدم لوصف ال encoding كما هو في ال LPCM.

3. Differential Pulse Code Modulation (DPCM):

الفكرة من هذه الخوارزمية أنه بدلاً من أن نرسل الاشارة كاملة نقوم أولاً بإرسال أول قيمة بالإشارة ثم نرسل الفرق بين القيمة التالية والقيمة الحالية بدلاً من إرسال القيمة التالية كاملة, تفيد هذه العملية بتقليل حجم data لأن الفروقات أصغر من الأرقام الاصلية, وبالتالي تتمثل ببتات أقل وكما أن التكرار في الفروقات كبير.

مثلاً:

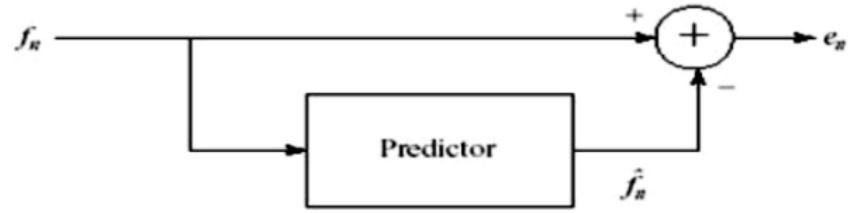
لدينا الاشارة التالية 140,130,120,110 عند تطبيق DPCM عليها تصبح:

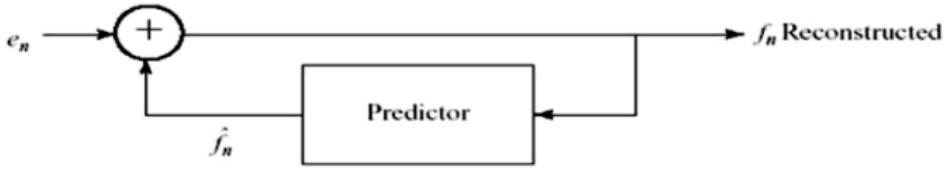
110, (120-110), (130-120), (140-130)

110,10,10,10

نلاحظ أن الارقام قد صغرت وأصبحت مكررة.

- يوجد طريقة أخرى لتطبيق DPCM وهي أن نقوم بتوقع كل قيمة من قيم data بخوارزمية معينة ثم نقوم بإرسال الفرق بين القيمة الحقيقية والقيمة المتوقعة وهذه الطريقة أفضل لان الفرق بين القيمتين الحقيقية والمتوقعة غالباً يكون صغير جداً مما يجعل العدد صغير أيضاً.

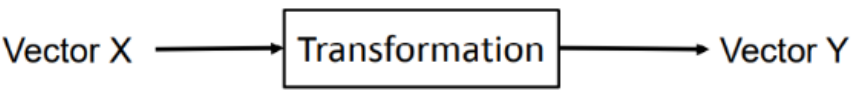




- يوجد نوعين للضغط باستخدام DPCM:
- Lossless: تشفيرencode ونقل الفروقات كما هي .
- Lossy: بهذا النوع نقوم بتكميم الفروقات التي أوجدناها إلى مستويات محددة قبل تشفيرها ونقلها, مثلاً نحدد 5 مستويات (50,40,30,20,10)ونقوم بتقريب أي فرق ينتج لدينا إلى أحد هذه المستويات،أي إذا نتج فرق يساوي 38 نقربه إلى المستوى 40 وهكذا ..نكون قد خسرنا القليل من data ولكن حصلنا على ضغط أعلى بسبب التكرار الكبير للقيم.

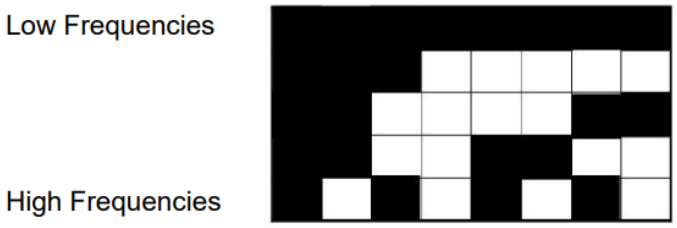
Transform Coding

- تقوم هذه العملية بنقل الإشارة من فضاء X إلى فضاءY (مثل الفضاء الترددي أو..) ثم نطبق عملية الضغط في ذلك الفضاء، وسبب قيامنا بهذا هو أنه يمكن أن تكون المعلومات في الفضاءY قابلة للضغط أكثر من الفضاء X (محاولة تشفير Y بدل X إذا كانت مكونات Y أقل ارتباطًا بكثير من مكونات X ، فيمكن ترميز Y بشكل أكثر كفاءة من X):



وعندما نريد فك الضغط نقوم أولاً بفك الضغط عن الإشارة ثم نرجعها إلى الفضاء X بالمعادلة العكسية للتحويل.

- إن الفضاءات التي لا ننقل لها الاشارة هي الفضاءات التي تعتمد على التكرار Frequency حيث أن في الصور يكون frequency منخفض عندما لا يكون هنالك تغيير كبير في البكسلات (أجزاء مهمة من الصورة و تكميمها خفيف : أي ضغطها أقل) ، أما frequency العالي يكون عندما تتغير البكسلات بشكل كبير(تمثل تفاصيل الصورة وهي أقل أهمية , تكميمها أكبر : ضغطها أعلى).



- من هذه التحويلات التي نطبقها على الاشارة لنقلها لفضاء آخر هو تحويل DCT.

DCT (Discrete Cosine Transformation)

هو تحويل بسيط يشبه تحويل فورييه لكنه أبسط حسابياً ولكن كلاهما يمتلك الفكرة الرياضيّة نفسها وهي تحويل الاشارة إلى مجموعة من إشارات sin، الفرق بين DCT و DFT أنه في DCT الناتج هو أعداد حقيقيّة أما في DFT فالناتج أعداد عقديّة.

معادلة DCT:

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

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

- حيث أن F(0) تسمى DC coefficient وباقي القيم F(2),F(1)..... تسمى AC coefficients .

العلاقة العكسية لل DCT :

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

معادلات DCT للإشارة 2D (صورة):

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

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

حيث:

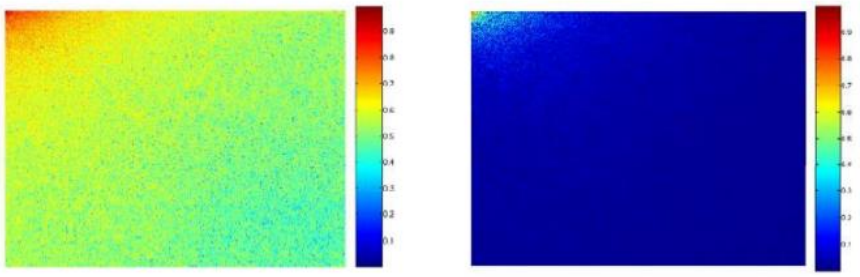
$$C(k) = \begin{cases} \frac{1}{\sqrt{2}} & for\ k = 0 \\ 1 & otherwise \end{cases}$$

التحويل المعاكس ل DCT للإشارة 2D (صورة):

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

$$m = 0,1, \dots \dots, N-1 \qquad n = 0,1, \dots \dots \dots, N-1$$

مثال:



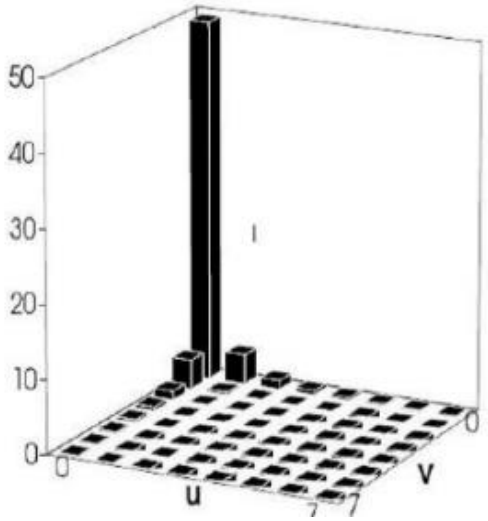
DFT DCT

نلاحظ أن:

تحويل DCT تكون الإشارة مجمّعة في الزاوية (القيم الخضراء في الزاوية العليا اليسارية) وبقية الصورة هي عبارة عن قيم صغيرة (القيم الزرقاء)وهذا يفيد أن Data المهمة والتي تعبر عن الإشارة كاملة محصورة في الزاوية اليسارية العليا والقيم البقية هي قيم صغيرة قريبة من الصفر نعتبرهم صفر.

أما تحويل DFT نلاحظ أن الاشارة ليست مركزة في الزاوية تماماً بل موزعة على قسم كبير من الصورة.

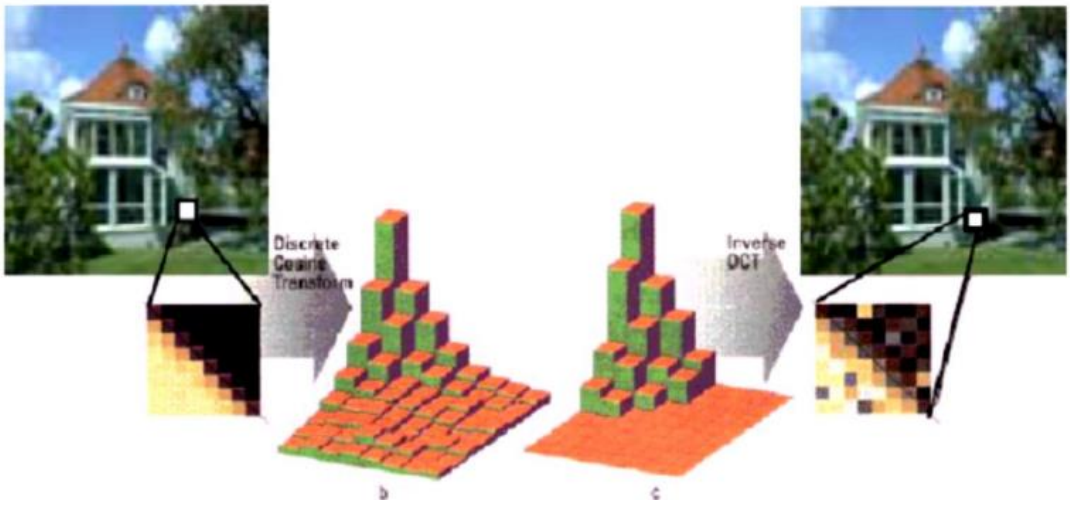
- وعند تمثيل DCT بشكل ثلاثي الابعاد ينتج لدينا الشكل الاتي :



حيث تمثل القيمة الكبيرة التي في الزاوية DC F(0) وباقي القيم تسمى AC .

- وعادة عندما نستخدم DCT على الصور نقوم بتقطيع الصورة إلى عدد من Blocks بحيث يكون أبعاد كل Blocksهو 8*8 .

مثال:



نرى في الصورة السابقة أننا أولاً أخذنا block من الصورة الاصلية اليسارية ثم طبقنا عليها تحويل DCT فنتج المخطط "b" ثم قمنا بتصفير القيم الصغيرة في DCT فنتج المخطط "c" وهنا نكون قد خسرنا معلومات, ثم طبقنا تابع DCT العكسي IDCT فعدنا للصورة الاصلية مع خسارة بعض التفاصيل فيها خصوصاً عند الحواف.

وهنا يمكن أن نسأل كيف نحدد ما هي القيم التي يجب تصفيرها؟؟

هناك جدول التكميم وهو جدول إحصائي ثابت بحيث سوف أقوم بعملية التكميم على هذا الجدول.

مثلاً في صور JPEG نحولها الى DCT ثم نقوم بتقسيم كل قيمة على قيمة من الجدول وهكذا نستطيع معرفة ماهي القيم التي يجب تصفيرها وسوف يتم شرحها اكثر لاحقاً.....

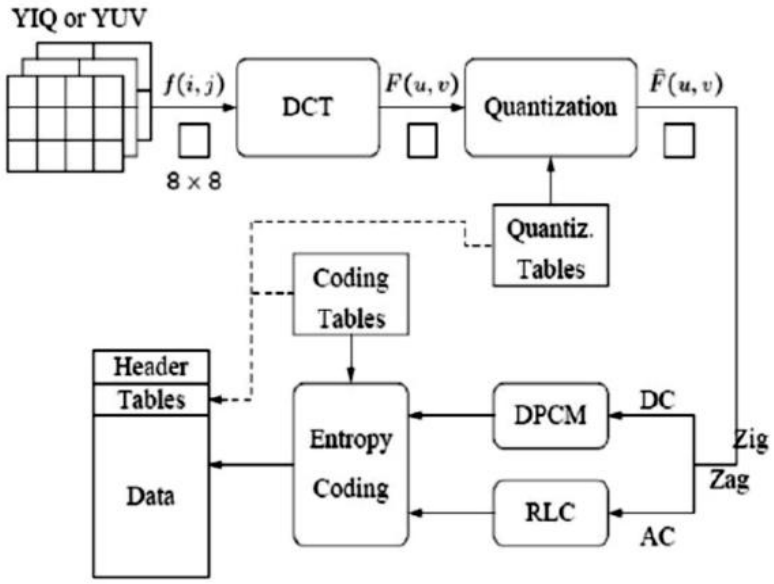
16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

JPEG Compression

خوارزمية معتمدة من 1992,وتستخدم عدة parameters تسمح للمستخدم بتحديد كمية الداتا التي ستضيع أثناء الضغط، عادةً العين لا يمكنها ملاحظة الضغط حتى إلى النسب الآتية: 10:1 و 20:1.

JPEGهي طريقة للضغط (lossy or lossless) للصور الثابتة الملونة أو ذات التدرج الرمادي.

تتألف من مجموعة من الخطوات وكل خطوة عبارة عن خوارزمية:



Steps Involved:

1. Discrete Cosine Transform of each 8x8 pixel array $f(x,y) \rightarrow_T F(u,v)$
2. Quantization using a table or using a constant
3. Zig-Zag scan to exploit redundancy
4. Differential Pulse Code Modulation(DPCM) on the DC component and Run length Coding of the AC components
5. Entropy coding (Huffman) of the final output

والآن سنشرح كل خطوة على حدىً.....

1- تحويل للفضاء اللوني (Converting to luminance/chrominance color space):

التحويل إلى فضاء ألوان يتعامل مع الإضاءة luminance بشكل منفصل عن الألوان chrominance

لأننا سوف نقوم بضغط الإضاءة بنسبة مختلفة عن نسبة ضغط الألوان حيث أن عين الإنسان حساسة لتغييرات الإضاءة أكثر من تغييرات اللون لذلك نريد ضغط الألوان أكثر, ونظام RGB لا يؤمن هذه الخاصية، لذلك نحول الصورة من فضاء RGB إلى فضاء يتعامل مع الإضاءة بشكل منفصل وهذه الخطوة هي خطوة إختيارية ولا تستعمل مع صور grayscale.

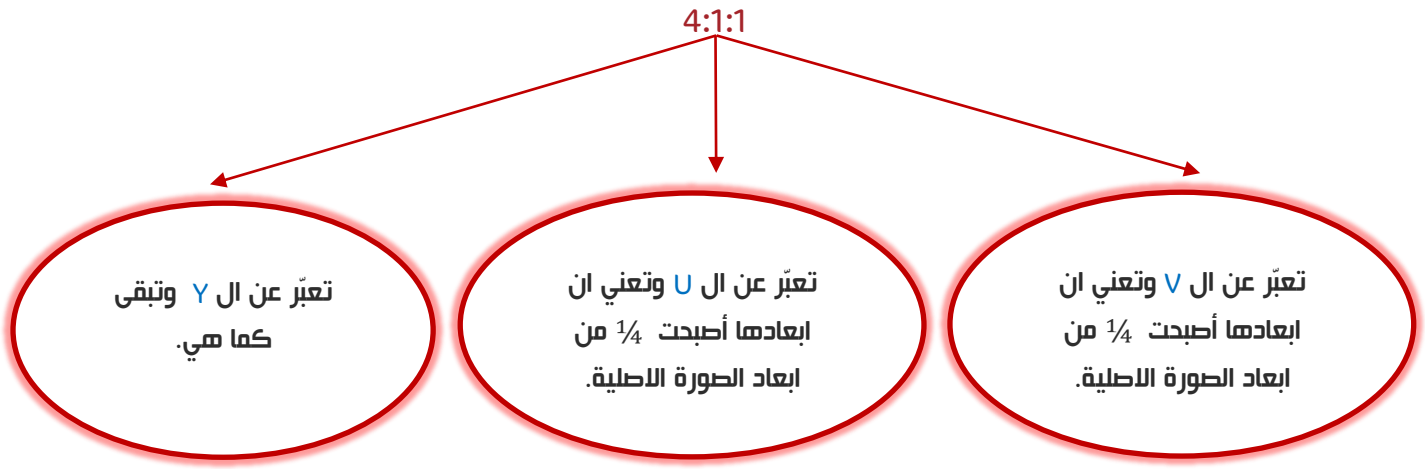
2- Subsampling

Chrominance components are subsampled (create low resolution images)

نقوم بعملية subsampling أي تخفيض ابعاد الصورة التي لدينا (أي نخفض دقتها) ونقوم بتطبيق هذه العملية على الألوان chrominance ولا نطبقها على الإضاءة luminance.

فمثلاً إذا حولنا الصورة الى نظام YUV , يكون لدينا 3 صور وهي ال Y وال U وال V ولهم نفس الأبعاد, فنقوم بعملية ال subsampling على ال U وال V لانهم الوان ولا نطبقها على ال Y لأنها اضاءة , فتصبح صور ال U وال V أبعادها أصغر من ابعاد الصورة الاصلية وتبقى ابعاد ال Y مساوية لأبعاد الصورة الاصلية.

- عادة تكون نسب ال subsampling بالشكل الآتي :



- من الممكن أيضاً أن تكون : 4:2:2 وعموماً لا نطبق هذه الخطوة على الصور ال grayscale .

3- Blocks (Create data units (8x8 pixels))

نقوم بتقسيم الصورة الى عدّة blocks , بحيث يكون ابعاد كل block هي 8*8, حيث ستقوم الخوارزمية بمعالجة كل block لوحده.

وهنا اذا كانت الصورة الاصلية غير قابلة للتقسيم على 8 نقوم بإضافة أعمدة واسطر حتى تصبح قابلة للتقسيم على 8 , وعند فك الضغط يجب أن نحذف هذه الاعمدة والاسطر الزائدة.

لدينا هنا طريقتين للمعالجة :

1. Non-interleaved mode

ونقوم هنا بمعالجة كل ال blocks للمركبّة الأولى Y ثم نقوم بمعالجة كل ال blocks للمركبّة الثانية U ثم نعالج كل ال blocks للمركبّة الثالثة V .

2. Interleaved mode

نقوم هنا بمعالجة ال block الأول للمركبّات الثلاث معاً ثم ال block الثاني للمركبّات الثلاث معاً وهكذا .

4- DCT (Apply DCT for data units)

نقوم بتطبيق الـDCT على كل ال blocks, وهنا تصبح طاقة الإشارة لكل block مجمّعة في الزاوية العلوية اليسارية.

وهنا يوجد خسارة لبعض ال data لأننا عندما نستخدم دالة ال Cos في الحساب تكون لدينا الدقة الحاسوبية محدودة.

5- التكميم Quantization

نقوم بعمل تكميم لخرج مرحلة ال DCT حيث نقرّب الأرقام الصادرة عن مرحلة ال DCT الى مستويات تكميميّة محدودة.

حيث لإيجاد القيم المكممّة نقوم بتقسيم كل رقم من خرج ال DCT (أي كل من ال 8*8=64 مركبة ترددية)على رقم معيّن يسمى ب QC (Quantization Coefficient) ثم نقوم بتقريب النتيجة الى اقرب عدد صحيح Integer.

ولدينا نوعين من التكميم :

1. Uniform Quantization:

حيث يكون ال QC ثابت لكل الاعداد.

2. Non-Uniform Quantization :

يكون ال QC مختلف لكل قيمة، بحيث يكون صغير للقيم التي في الزاوية العلويّة اليسارية لأنها أكثر أهمية.

ونقوم بإنشاء جدولين لقيم ال QC , جدول خاص بالإضاءة ، جدول خاص بالألوان , ونحسب قيمة كل رقم بالمعادلة التالية : $F'(u,v) = Round\left(\frac{F(u,v)}{Q(u,v)}\right)$

حيث: $F(u,v)$ هو القيمة الأصلية لل DCT coefficient.

$F'(u,v)$ هو القيمة لل DCT coefficient بعد التكميم.

$Q(u,v)$ قيمة التكميم.

The Chrominance Quantization Table

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

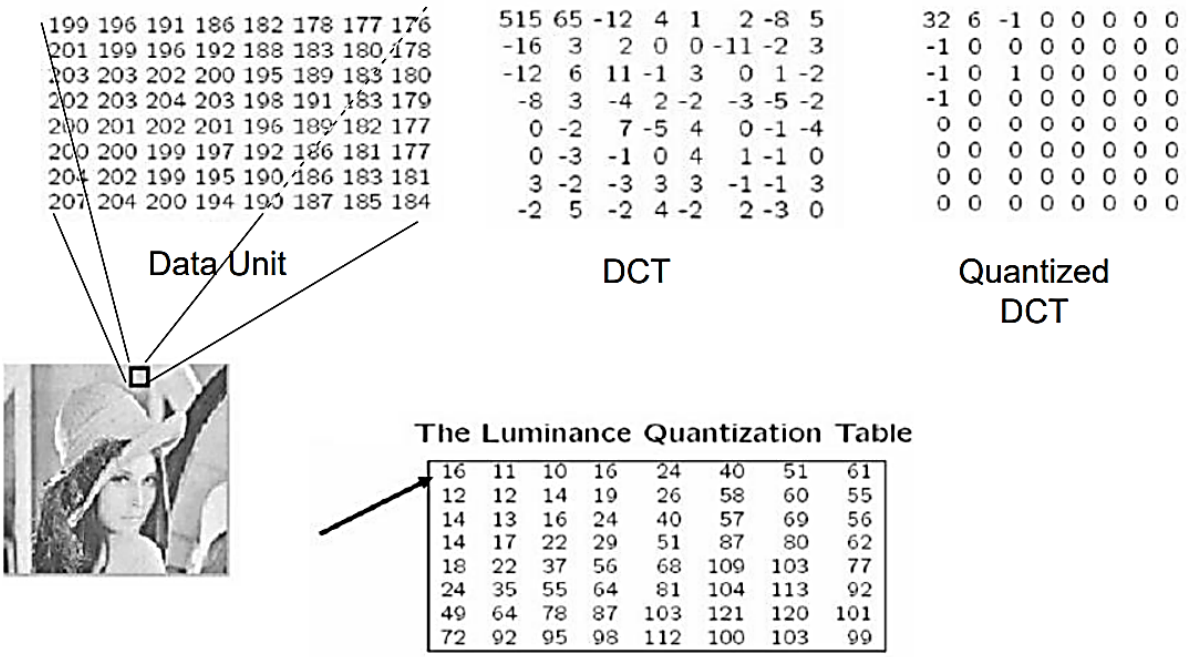
The Luminance Quantization Table

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

نلاحظ في الصورة السابقة في جدول QC للألوان Chrominance وجود الرقم 99 للقيم البعيدة عن الزاوية العلويّة اليسارية , وذلك لأننا لا نهتم لتلك القيم , فخسارتها لن تؤثر على الصورة بشكل كبير لذلك نكبر قيم ال QC لها ونلاحظ أن ال QC للعناصر في الزاوية اليسارية العليا صغيرة نسبيا وذلك لانها تحوي على data مهمة وخسارتها سوف يؤثر كثيرا على الصورة .

ومصفوفات ال QC هذه هي مصفوفات Standard , أي معروفة ومعمّمة , ويمكننا تخزينها في ال Header الخاص بالصورة.

لنأخذ مثال عن التكميم : نلاحظ هنا أن المصفوفة بعد التكميم اصبح معظمها اصفار , والقيم غير الصفرية هي قيم صغيرة نسبيا .



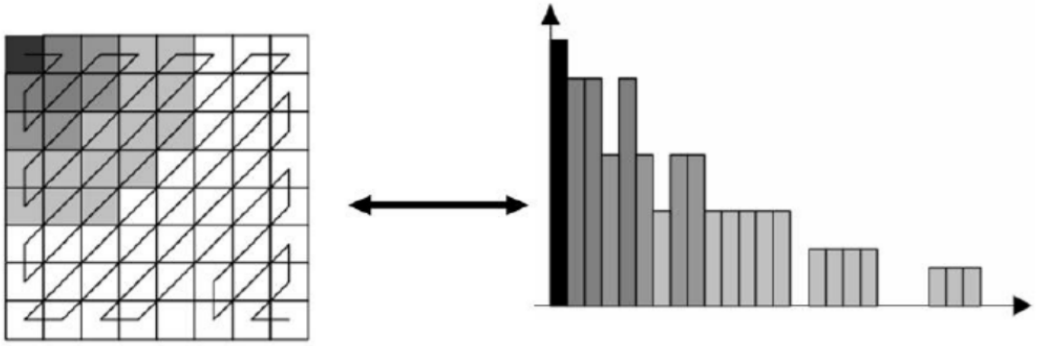
The Luminance Quantization Table

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

6- Zigzag (Zig Zag Quantized DCT to get 64 vector) :

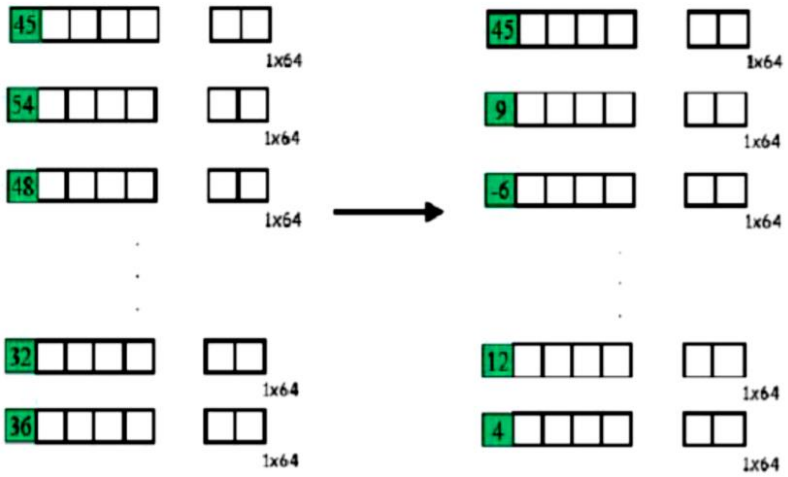
عندما نريد ارسال المصفوفة التي نتجت لدينا بعد التكميم , لا نقوم بإرسالها سطر سطر, لأننا نريد اكبر عدد من الاصفار المتتالية ليكون الضغط فعّال , وبما أن القيم غير الصفرية كلها متجمعّة في الزاوية اليسارية العليا , نقوم بإرسال ال data على شكل zigzag بدءاً من الزاوية اليسارية العليا.

Zig Zag for DCT coefficients



7- Coding :

بعد مرحلة ال zigzag يكون قد نتج لدينا vector مؤلف من 64 عنصر، فنقوم في هذه المرحلة بتطبيق ال DPCM على اول قيمة من كل vector والتي هي ال DC , حيث نقوم بطرح قيمة ال DC في ال vector الحالي من قيمة ال DC في ال vector التالي , فينتج لدينا ارقام صغيرة نسبيا , وسبب ذلك هو أن الألوان بين كل vector و vector (أي بين كل block و block) قريبة من بعضها ولذلك تكون قيم ال DC لها قريبة ايضاً.



8- Entropy Coding :

عندما نريد تمثيل ال DC بعد أن طبقنا عليه ال DPCM نقوم بتمثيله بقيمتين : (Size,Value) , حيث :

Size : يمثل عدد البتات اللازمة لتمثيل قيمة ال DC , Value : قيمة ال DC بالبتات .

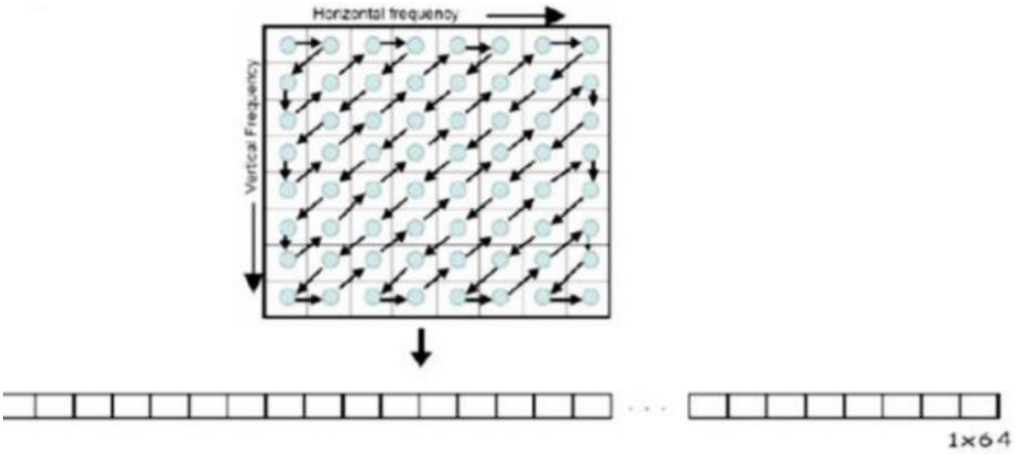
حيث نقوم بإنشاء جدول لتمثيل هذه القيم مثل الجدول الآتي :

SIZE	Value	Code
0	0	---
1	-1,1	0,1
2	-3, -2, 2,3	00,01,10,11
3	-7,..., -4, 4,..., 7	000,..., 011, 100,...111
4	-15,..., -8, 8,..., 15	0000,..., 0111, 1000,..., 1111
.		.
.		.
11	-2047,..., -1024, 1024,... 2047	...

Size and Value Table

-8 is encoded as: 1010111
101: 4 (the size)
0111: -8 (The value)

9- RLE on AC components (after Zig-Zag Scan)

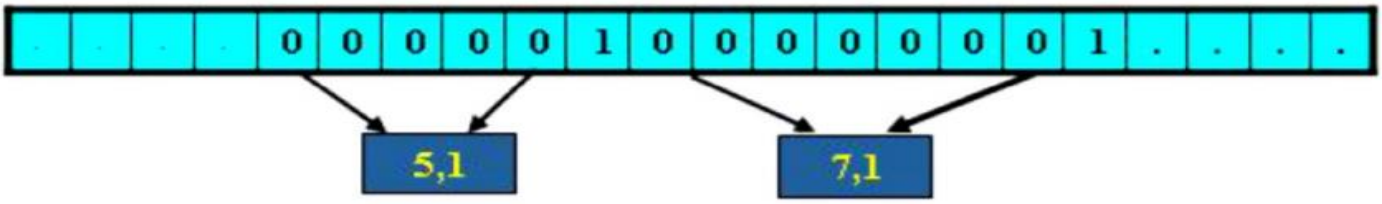


نقوم بتطبيق خوارزمية ال RLE على ال 63 عنصر التي بقيت في ال vector والتي تسمى بال AC , فنقوم بترميزهم بالشكل الآتي : (Skip,Value)

حيث :

Skip : عدد الاصفار المتتالية

Value : هو الرقم غير الصفري الذي يتبع سلسلة الأصفار.



وعند نهاية ال vector نرسل الثنائية (0,0) لتدل على أن ال vector قد انتهى .

10- Entropy Coding on RLE on AC Components

نقوم هنا بتحديد عدد البتات اللازمة لتمثيل العنصر غير الصفري فيصبح التمثيل بالشكل : (S1,S2)

- S1:
  - RunLength: length of zeros (مؤلفة من عدد الأصفار المتتالية)
  - Size: number of bits needed for the next non-zero component (وعدد البتات اللازمة لتمثيل العنصر غير الصفري)
- S2: • The value of the AC component (refer to size and value table)

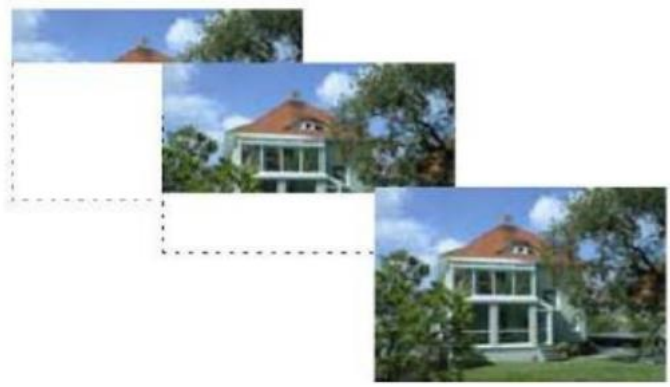
هكذا تكون انتهت مجموعة خطوات خوارزمية ضغط الصور JPEG 😂

JPEG Model

هنا نهتم بطريقة تحميل وعرض الصور عندما نفتح صفحة Web مثلاً ولها عدّة طرق :

1. Sequential Mode

وهنا يتم إجراء عملية ال encoding & decoding للصورة مرة واحدة، ويتم إجراء ال encoding بمسحة واحدة من اليسار إلى اليمين ومن الأعلى إلى الأسفل.



2. Progressive Mode

تعتمد هذه الطريقة على إظهار الصورة كاملة بدقة قليلة حيث تظهر الصورة بشكل مشوّش blurry أولاً ثم تقوم بتحميل المزيد من ال data فتتوضّح الصورة أكثر.

عند ضغط الصورة بهذه الطريقة يقوم ال encoder بضغط المعلومات الأكثر أهميّة من الصورة اولاً، ثم يضع بعدها المعلومات الأقل أهمية كي يستطيع العارض إظهار الصورة كلها بدقة منخفضة اثناء الفك أولاً وعند انتهاء الفك تصبح الصورة واضحة ،تستخدم عند إرسال الصور المضغوطة عبر الشبكة ليتم فك ضغطها بال real time ,مدعومة بشكل خاص في المتصفحات browsers.

(a) (b) (c)

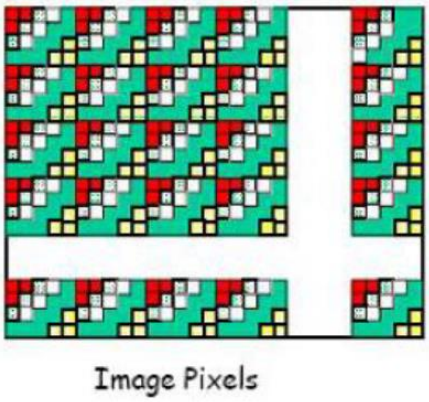
ولهذه الطريقة نوعين:

- Spectral selection:

وهنا يتم عمل encode للصورة بوضع الـ DC لكل الـ blocks المؤلفة للصورة أولاً.

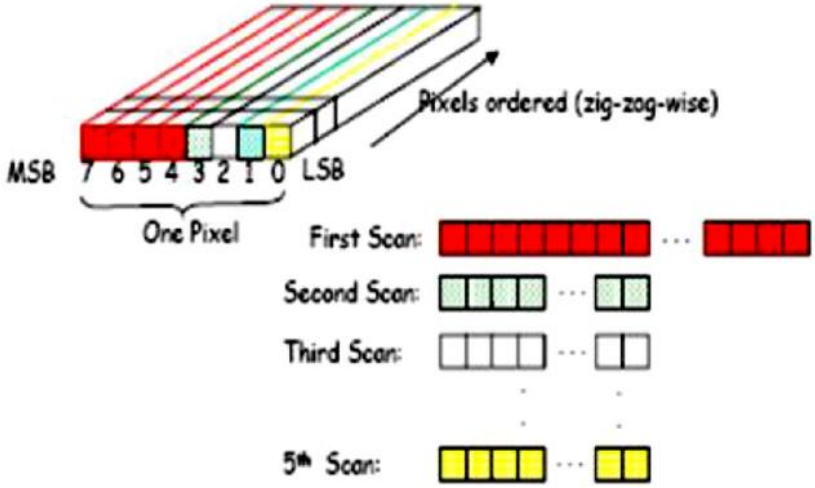
ثم نضع قسم من الـ AC لكل الـ blocks ،ثم قسم ثاني وهكذا إلى أن نصل إلى نهاية كل الـ blocks ,لذلك تتم العملية على عدة مسحات scans .

وعندما نفك الضغط عن الصورة نقوم أولاً بأخذ قيم الـ DC ونملأ باقي الصورة أصفار ونعرضها، فتكون الصورة blurry ، ثم نأخذ ثاني مجموعة من أرقام الـ AC ونضيفها للصورة فتتوضح أكثر وهكذا...



- Successive Approximation

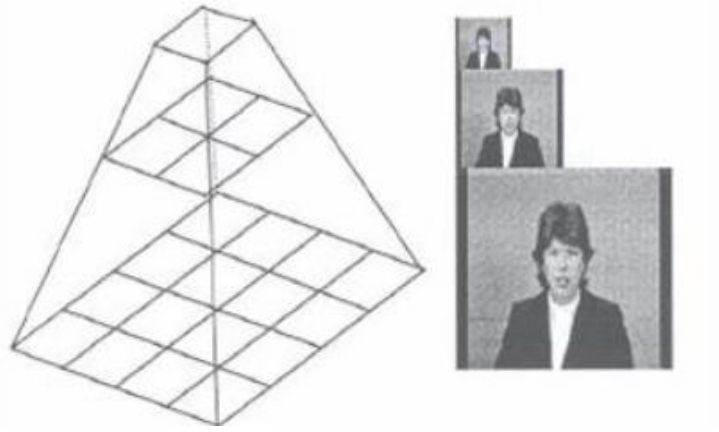
هنا نقوم بإرسال عدة بتات من ال DCT في المرة الواحدة فمثلاً نقوم بإرسال أول 4 بتات بدءاً من الـ bit الأكثر أهمية MSB من كل الـ bytes في كل block، ثم نضيف ثاني 4 bits وهكذا..



الرجاء الاطلاع على السلايد 45

3. Hierarchical Mode

وهي طريقة بديلة عن ال progressive, هنا نخزن بالصورة نفسها عدّة نسخ من نفس الصورة، لكنها أصغر وعند فك الضغط نفك الضغط عن الصورة الصغيرة أولاً ونعرضها بحجم الصورة الأصليّة فتظهر بدقة منخفضة ثم نظهر الصورة الأكبر فالأكبر.



-انتهت المحاضرة-