تشفير صورة باستخدام خوارزمية جديدة لتحديد وتشفير حواف ألوان الصورة

Image coding by new coding edges colors image Algorithm

هند رستم محمد
حسن ثابت رشيد
جامعه الكوفة / كلية التربية / قسم الحاسبات

الخلاصة :

قدم البحث طريقة جديدة يمكن من خلالها تشفير صورة معتمداً على تحديد حواف أي لون في الصورة وتحديد عدد الألوان الخاصة بالصورة ,بعدها يتم تحليل القيم المستخرجة لتحديد الحواف لكي يتم تشفيرها بخوارزمية تبديل الأعمدة المستخدمة من خلال معامل (Laplacian operator) .

علماً إن عدد المفاتيح كان (24) مفتاح من مفاتيح التشفير ، كما تعطي الطريقة إمكانية فك التشفير للصورة ذاتها حيث يمكنها التعامل مع كل أنواع الصور ومختلف إحجامها إذ تقوم الإلية المعتمدة على تحويل أي نوع من الصور المراد تشفيرها أو فك التشفير إلى الصور ذات الامتداد (BMP.) وتقسم الطريقة إلى :

- مرحلة تحديد نوعية الصورة وحجمها وحواف الوانها
- مرحلة تحليل الصورة
- مرحلة المعالجة

تم استخدام حاسب نوع بنتيوم 4 , 1700 GHz, لغة دلفي بإصدارها الخامس وقد أثبتت نتائج المحاكاة جودة توفيق النموذج ألمحاكاتي المبتكر في إجراء معالجة التشفير وفك التشفير بدرجة 100% .

Abstract:

New method for coding and encoding color image is performed by selection Algorithm for edges colors image , analysis it values with laplacain operator than coding by one of 24 keys from coding keys. The programmer can input any types and any size of image by this method, to convert the type to (.BMP)image type.

The simulation contain three basic stages

1-Determent of the type and size image stage and edges colors

2-Analysis image stage

3-Processing image stage

Pentium 4,1700GHz,Delphi language used to implementation this simulation the results show 100% of correct coding and encoding image.

المقدمة

يعتبر مجال معالجة الصور الرقمية فرعاً من فروع الذكاء الاصطناعي التي توفر إمكانية عرض الصور الرقمية بدقة ومرونة عالية حيث اخذ هذا الفرع عمقاً كبيراً في اختصاصات عدة تتطرق لمفاهيم ومحتويات في أكثر من مجال اعتماداً على كفاءتها في تطبيق معين ,فقد كانت التطبيقات الأولى لمعالجة الصور الرقمية في عام 1921م ، وكانت تعمل على تحسين الصور للصحف المحمولة إلى ألصيغه الرقمية والمرسلة بكابل بحري بين لندن ونيويورك [1] .

وبعد ظهور الحاسبات الاليكترونية بدا في عام 1964م استخدام الحاسبة في علوم الفضاء واستخدام الصورة المأخوذة للقمر والمرسلة في مجسات الفضاء بواسطة مختبر (jet prooulsuion) في (Pasadena) من ولاية كاليفورنيا خلال مركبة الفضاء (ranger 7) علماً إن ألصوره كانت تعاني من تشوهات ناتجة عن الموجة التلفزيونية المركبة على سطح المركبة الفضائية [2].

لقد ظهرت العديد من الدراسات والبحوث في مجالات استخدام الصورة الرقمية بالحاسبة الاليكترونية إلى إن ظهرت الحاجة إلى وجود إمكانية نقل الصورة من مصدر إلى أخر بدون إن تكتشف وبشكل مشفر[3], وكان للصورة من النوع (BMP) ذات أفضلية في النقل من حيث الأمنية حيث تمتاز بالجودة العالية في العرض وإمكانية التحرير والعرض بسهولة فائقة مع برامج العرض المتنوعة بالإضافة إلى انه لا يوجد هناك فقدان لبيانات هذه الصور إثناء المعالجة ،ولأنها تأخذ إحجاماً كبيرة بالخزن فإنها تكون قليلة الاستخدام والعرض من خلال الانترنت [4].

مراحل النظام:

تتمثل بالتالي :

1)مرحلة تحديد نوعية الصورة وحجمها :

نوعية الصورة

حيث يتم تحويل الصورة المراد تطبيق النظام المقترح عليها من أي نوع كانت إلى صورة نقطية (أي صورة ذات امتداد من نوع (BMP.) وذلك لامكانيه التعامل المباشر مع محتويات هذا النوع من الصور حيث أنها لا تخضع لأي خوارزميات ضغط أو تشفير للبيانات الاصليه .

حجم الصورة

الصورة المطبقة لا تقتصر على حجم معين أو محدد حيث يمكن استعمال صورة بأي حجم كانت سواء صغيرة أو كبيرة .

الإحجام التي طبق عليها النظام

125*125
400*300
750*600
800*800
1024*1024

معامل Laplacian operator

توجد ثلاث نوافذ (Masks) خاصة بهذا النوع من العمليات أثبت كفاءته في تحديد حواف الصورة وتحديد حواف كل لون في الصورة ,حيث إن مجموع معاملات أي ماسك هنا تساوي (0) أي يساعدنا على تحديد خلفية سوداء وتوضيح خواص اللون المستخرج وبيان حوافه بشكل دقيق[2,6] .

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 1 & -2 & 0 \\ -1 & 4 & -1 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

والجزء البرمجي الخاص به هو:

```
for I:=1  to x do
  for j:=1  to y do
   begin
    k[0]:=image1.canvas.pixels[i-1,j-1]* -1;
    k[1]:=image1.canvas.pixels[i-1,j]*-1;
    k[2]:=image1.canvas.pixels[i-1,j+1]*-1;
    k[3]:=image1.canvas.pixels[i,j-1]*-1;
    k[4]:=image1.canvas.pixels[i,j]*8;
    k[5]:=image1.canvas.pixels[i,j+1]*-1;
    k[6]:=image1.canvas.pixels[i+1,j-1]*-1;
    k[7]:=image1.canvas.pixels[i+1,j]*-1;
    k[8]:=image1.canvas.pixels[i+1,j+1]*-1;
      sum:=0;
     for n:=0 to 8 do
      sum:=sum+ round(0.5*k[n]);

image1.canvas.pixels[i,j]:=round(sum);
```

2)مرحلة تحليل الصورة :

مرحلة الحصول على بيانات الصورة وتحليلها أو تحويلها إلى الصيغة أو الشكل الذي من خلاله نستطيع تطبيق النظام المقترح عليها , ونستطيع تلخيصها بالعمليات الآتية :

أ- تحليل الصورة وإجراء عملية مسح (SCANNING) لها للحصول على بيانات كل عنصر عرض في الصورة (Pixel) وعادة ما تكون موضوعه بالنظام العشري (Decimal) .

ب- نقوم بعملية تحويل صيغة بيانات الصورة من النظام العشري (Decemal System) إلى النظام الثنائي (Binary System) . حيث إن كل عنصر (Pixcel) في الصورة يتم تحويله إلى 24 بت في النظام الثنائي , وبما إن كل عنصر عرض يتكون من ثلاث ألوان أساسية مشتركة في تكوينه وهي الأحمر,الأخضر,الأزرق (نظام ألوان RGB). فان لكل لون من هذه الألوان يحجز 8 بتات خاصة به وبالنتيجة فان الشدة اللونية لكل لون سوف تتراوح ما بين (0_ 255).

جـ - يتم خزن بيانات الصورة لكل عنصر عرض (Pixel) نقوم بخزنها في مخزن وسطي (Buffer Image) وذلك لاستخدامه في المراحل اللاحقة . وجزء المعالجة البرمجي الخاص بهذه المرحلة هو :

```
for I:=0  to image1.picture.height-1 do
for j:=0  to image1.picture.width-1 do
begin
z1:=image1.canvas.pixels[i,j];
for k:=1 to 24 do
 begin
if z1 mod 2=0 then
begin
if k <= 8 then
r[i,j]:=r[i,j]+'0'
else if (k <= 16) then
g[i,j]:=g[i,j]+'0'
else   b[i,j]:=b[i,j]+'0';
end else begin
if k <= 8 then
r[i,j]:=r[i,j]+'1'
else if  (k<=16) then
g[i,j]:=g[i,j]+'1'
else  b[i,j]:=b[i,j]+'1';
end;
z1:=z1 div 2;
end;
s1:=r[i,j]+g[i,j]+b[i,j];
end;
```

3)مرحلة المعالجة :

في هذه المرحلة نقوم بإجراء عملية المعالجة الخاصة بالتشفير ويمكن توضيحها بالخطوات التالية :

أ- مرحلة المعالجة الابتدائية :

يتم في هذه الخطوة سحب بيانات كل عنصر عرض في الصورة وتحويله إلى مصفوفة ثنائية بأربعة صفوف وستة أعمدة (4*6) . والجزء البرمجي الخاص بهذه الخطوة :

```
W:=1;
for f1:=1 to 4 do
for f2:=1 to 6 do
begin
h1[f1,f2]:=s1[w];
inc(w);
end;
```

ب- مرحلة المعالجة الوسطية :

تعتبر أهم مرحلة في النظام حيث إن كفاءة النظام ونجاحه تعتمد على هذه المرحلة وقدرتها على تشفير الصورة ,إن الطريقة المتبعة هي عملية إبدال ما بين أعمدة أو صفوف هذه المصفوفة وبالتالي الحصول على خلطة لونية جديدة مختلفة تماماً عن الخلطة اللونية الخاصة بالصورة الأصلية وبالنتيجة فانه لايمكن التعرف على هذه الصورة الجديدة .

يمكن توضيح عملية المعالجة هذه بالحالات الآتية :

1. إن المصفوفة الثنائية الخاصة بكل عنصر عرض في الصورة ذات حجم 6*4 فانه يمكن قراءتها بإحدى الحالتين:
   - قراءتها بصيغة صف _ عمود
   - قراءتها بصيغة عمود _ صف

وفي كلتا الحالتين فانه هناك نتيجة تشفير مختلفة للصورة المشفرة الناتجة .

2. إجراء عملية الإبدال (Swapping) وتوجد حالتين هما:
   - إبدال ما بين الأعمدة الخاصة بالمصفوفة
   - إبدال ما بين الصفوف الخاصة بالمصفوفة

وفي كلتا الحالتين فانه يوجد عملية تشفير مطبقه تختلف نتائجها وكفاءتها في تطبيق النظام , والجزء البرمجي الخاص بهذين الخطوتين :

```
for f1:=1 to 4 do
for f2:=1 to 6 do
hc1[f1,f2]:=h1[f1,strtoint(edit1.text[f2])];
```

جـ - مرحلة المعالجة النهائية وإظهار النتائج :

عملية استرجاع البيانات بعد تشفيرها ووضعها في الصورة الجديدة المشفرة وتتم بالخطوات التالية :

1)إعادة مصفوفة العنصر الثنائية إلى مصفوفة أحادية والجزء البرمجي الخاص بها هو :

```
for f1:=1 to 4 do
for f2:=1 to 6 do
S2:=S2+hc1[f1,f2];
```

2) تحويلها من النظام الثنائي إلى النظام العشري والجزء البرمجي الخاص بها هو :

```
for k:=1 to 24 do
begin
if s2[k]='1' then
ss:=ss+p;
p:=2*p;
end;
```

3)وضع هذه القيمة العشرية الجديدة لعنصر العرض في الجزء المخصص له في الصورة الجديدة(المشفرة) والجزء البرمجي الخاص به هو :

```
image2.canvas.pixels[i,j]:=ss;
```

إن مراحل تطبيق النظام التي تم توضيحها استخدمت في حالة إجراء عملية معالجة تشفير الصورة معينة بحجم معين غرض إرسالها إلى الشخص معين عبر الانترنت أو لحمايتها من الانتشار بين المتطفلين أو السرقة.

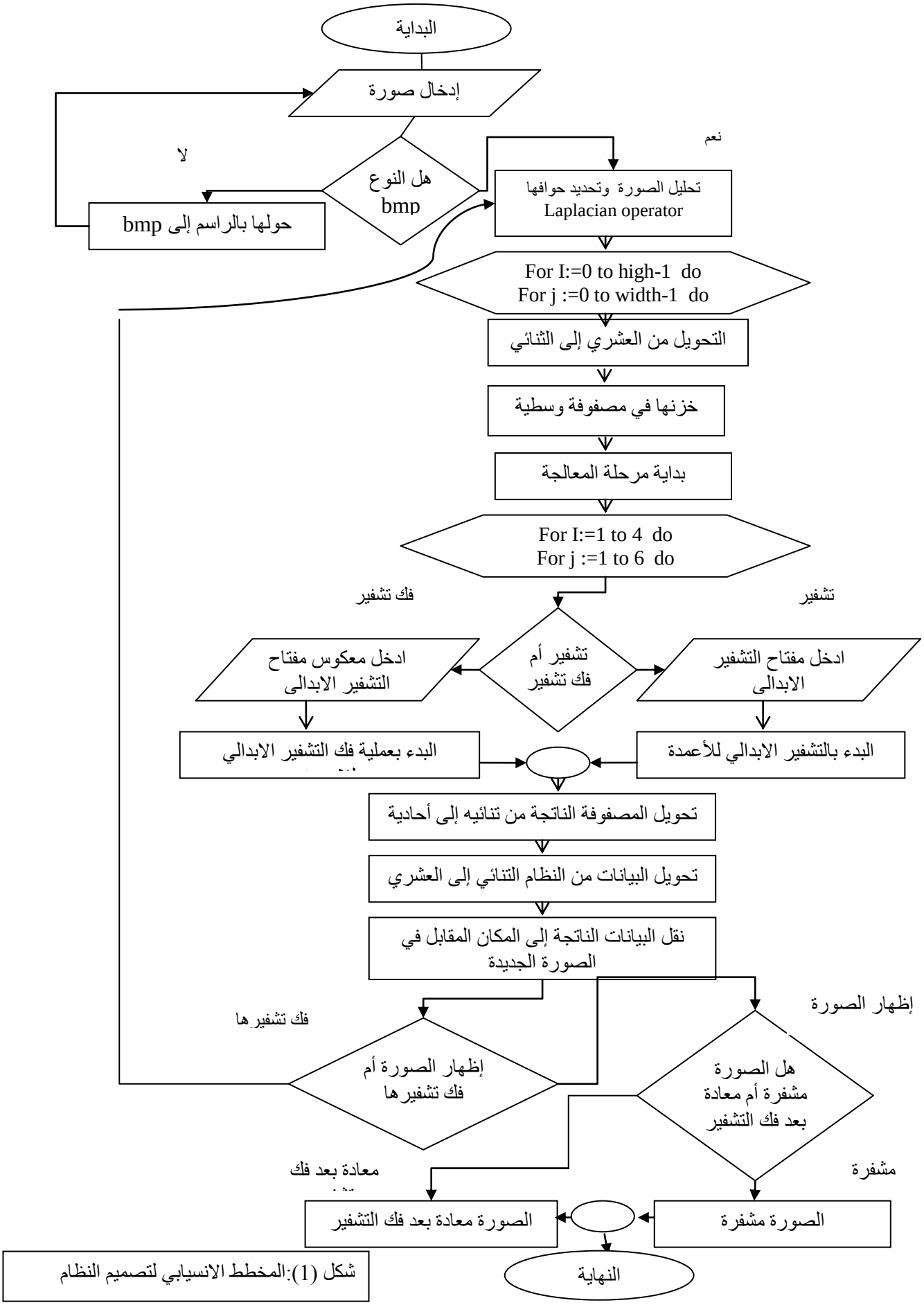
إن مراحل عملية فك التشفير لبيانات هذه الصورة واستعادة بياناتها الأصلية (الصورة المعادة بعد فك التشفير) هي نفسها مراحل النظام المتمثلة في

- مرحلة تحليل الصورة
- مرحلة معالجة الصورة(المعالجة الأولية , المعالجة الوسطية ,المعالجة النهائية)

مع فرق واحد هو استخدام معكوس المفتاح الابدالي الخاص بالتشفير وهذا المفتاح سيستخدم لفك التشفير والذي سنوضحه لاحقا في المناقشة .

تطبيق النظام والمناقشة :

تطبيق النظام: المخطط الانسيابي التالي يوضح مراحل تطبيق النظام المقترح :



شكل (1):المخطط الانسيابي لتصميم النظام

المناقشة :

تم تطبيق عملية الإبدال مابين الأعمدة وذلك لأفضليتها في إيجاد نتائج تكون أكثر تشويشاً أو عتمةً ,إي إيجاد تغيير ملحوظ
في الخلطات اللونية للصورة المشفرة الناتجة .

1. حجم المصفوفة 4*6 أي إن عدد الأعمدة هو 6 وبالتالي فان هناك 24 احتمال من الاحتمالات الابدالية التي ستجرى على أعمدة المصفوفة وكل احتمال سيقود إلى نتيجة تشفير معينة قد تكون ناجحة أو فاشلة (أي قد تغير من معالم الصورة الأصلية تغييراً كاملاً وبالتالي لايمكن التعرف على الصورة أو قد تبقى بعض من معالم الصورة وبالتالي فان عملية التشفير تكون غير كفوءة .
2. عدد الأعمدة هو 6 لذلك فان مصفوفة عنصر الصورة الأصلية تكون مرتبة من واحد إلى ستة كالأتي :

	Cul1	Cul2	Cul3	Cul4	Cul5	Cul6
Row1	1	2	3	4	5	6
Row2	7	8	9	10	11	12
Row3	13	14	15	16	17	18
Row4	19	20	21	22	23	24

ولذلك استخدمنا مفتاحاً خاصاً لذلك حجمه 6 من خلاله ستطيع السيطرة على عملية الإبدال ما بين الأعمدة، وسنطلق علية اسم مفتاح التشفير الابدالي ((Cipher key of swapping)) حيث يتم إدخاله كمعطى بياني قبل إجراء عملية الإبدال التي ستعتمد اعتماداً كلياً علية . وليكن قيمة المفتاح هي التالي :

2 4 1 5 6 3

من هذا المفتاح نستنتج بأنه سوف تطبق الابدالات التالية :

المصفوفة الأصلية قبل إبدال الأعمدة						المصفوفة الناتجة بعد إبدال الأعمدة					
1	2	3	4	5	6	2	4	1	5	6	3

3. إن استخدام معكوس المفتاح الابدالي ((Cipher key inverse of swapping)) الخاص بفك التشفير يمكن توضيحه كالأتي:

ترتيب الأعمدة في الصورة الأصلية هو :

1	2	3	4	5	6

ترتيب الأعمدة في مصفوفة التشفيـــر (مفتاح التشفير) :

2	4	1	5	6	3

ترتيب الأعمدة في مصفوفة فك التشفير (معكوس المفتاح) :

3	1	6	2	4	5

مثال يوضح خطوات تطبيق النظام

المدخلات :

الصورة النقطية ذات الحجم (125*125)
مفتاح التشفير الابدالي : يوجد (24) مفتاح سنطبق بعضاً منها بالإضافة إلى المفتاح

2	4	1	5	6	3

المعالجة : يمكن توضيحها بالجداول الآتية :

جدول(1) يوضح القيم العشرية لعناصر الصورة اللونية لمقطع حجمه (10*10) من الصورة ألمستخدمه في مثال التطبيق .

xels	Pix0	Pix1	Pix2	Pix3	Pix4	Pix5	Pix6	Pix7	Pix8	Pi
Pix0	14929332	14929332	14929078	15060664	14994871	14863285	15060405	15126198	14994867	149948
Pix1	14929587	14863539	14929078	14994871	14929078	14797492	15126198	14928819	14863281	150606
Pix2	14929332	14863539	14929078	14994871	14994871	14929078	15060405	14928819	14994867	151264
Pix3	14929332	14929332	14994871	15126457	15126457	15126711	14863026	15060405	15258039	151922
Pix4	14863539	14929332	14994871	15060664	15126711	15126711	14928819	14994867	15192246	152580
Pix5	14863539	14929332	14994871	14994871	14929332	14929332	15126453	14797488	14797488	151927
Pix6	14863539	14929332	14995125	14929332	14863539	14863539	15192246	14929074	14732208	150611
Pix7	14863539	14995125	15060918	14995125	14929332	14929332	15060660	15192759	15061173	148643
Pix8	14797746	14995125	14863539	14995125	14995380	15192759	15061173	15192759	15324345	149301
Pix9	14863539	14995125	14863539	14929332	14929587	15061173	15061173	14929587	15127480	147332

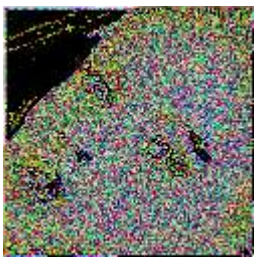
جدول(2) يوضح القيم الثنائية المناظرة للقيم العشرية لعناصر الصورة اللونية في الجدول أعلاه

pixel	Pix0	Pix1	Pix2	Pix3	Pix4	Pix5	Pix6	Pix7	Pix8	Pix9
Pix0	001011 011011 001111 000111	001011 011011 001111 000111	011011 010011 001111 000111	000111 010111 001110 100111	111011 011011 001100 100111	101011 011101 001101 000111	101011 011011 001110 100111	011011 010111 001101 100111	110011 011011 001100 100111	110011 011011 001100 100111
Pix1	110011 010111 001111 000111	110011 010011 001101 000111	011011 010011 001111 000111	111011 011011 001100 100111	011011 010011 001111 000111	001011 010101 001110 000111	011011 010111 001101 100111	110011 011101 001111 000111	100011 011101 001101 000111	001011 010111 001110 100111
Pix2	001011 011011 001111 000111	110011 010011 001101 000111	011011 010011 001111 000111	111011 011011 001100 100111	111011 011011 001100 100111	011011 010011 001111 000111	101011 011011 001110 100111	110011 011101 001111 000111	110011 011011 001100 100111	101011 011111 001101 100111
Pix3	001011 011011 001111 000111	001011 011011 001111 000111	111011 011011 001100 100111	100111 011111 001101 100111	100111 011111 001101 100111	111011 010000 101101 100111	010011 010101 001101 000111	101011 011011 001110 100111	111011 011000 101100 010111	011011 010000 101111 100111
Pix4	110011 010011 001101 000111	001011 011011 001111 000111	111011 011011 001100 100111	000111 010111 001110 100111	111011 010000 101101 100111	111011 010000 101101 100111	110011 011101 001111 000111	110011 011011 001100 100111	011011 010000 101111 100111	111011 011000 101100 010111
Pix5	110011 010011 001101 000111	001011 011011 001111 000111	111011 011011 001100 100111	111011 011011 001100 100111	001011 011011 001111 000111	001011 011011 001111 000111	101011 011111 001101 100111	000011 010101 001110 000111	000011 010101 001110 000111	111011 010100 101111 100111
Pix6	110011 010011 001101 000111	001011 011011 001111 000111	101011 010111 001100 100111	001011 011011 001111 000111	110011 010011 001101 000111	110011 010011 001101 000111	011011 010000 101111 100111	010011 010011 001111 000111	000011 011101 001100 000111	101011 010000 101110 100111
Pix7	110011 010011 001101 000111	101011 010111 001100 100111	011011 011111 001110 100111	101011 010111 001100 100111	001011 011011 001111 000111	001011 011011 001111 000111	001011 010111 001110 100111	111011 010100 101111 100111	101011 010000 101110 100111	001011 011111 001101 000111
Pix8	010011 011101 001110 000111	101011 010111 001100 100111	110011 010011 001101 000111	101011 010111 001100 100111	001011 011111 001100 100111	111011 010100 101111 100111	101011 010000 101110 100111	111011 010100 101111 100111	100111 010010 101110 010111	101011 010000 101111 000111
Pix9	110011 010011 001101 000111	101011 010111 001100 100111	110011 010011 001101 000111	001011 011011 001111 000111	110011 010111 001111 000111	101011 010000 101110 100111	101011 010000 101110 100111	110011 010111 001111 000111	000111 011100 101101 100111	101011 011111 001100 000111

جدول رقم (3) يوضح تقسيمات القيم الثنائية حسب نظام الألوان (RGB) المكونة من (24)بت والمناظرة لعناصر الصورة اللونية للجداول السابقة.

موقع البكسل في الصورة	القيمة العشرية للبكسل	القيمة الثنائية المناظرة للعشرية للبكسل			موقع البكسل بالصورة	القيمة العشرية للبكسل	القيمة الثنائية المناظرة للعشرية للبكسل		
		R	G	B			R	G	B
pixel (0,0)	14929332	00101101	10110011	11000111	pixel (5,0)	14863539	11001101	00110011	010001
pixel (0,1)	14929332	00101101	10110011	11000111	pixel (5,1)	14929332	00101101	10110011	110001
pixel (0,2)	14929078	01101101	00110011	11000111	pixel (5,2)	14994871	11101101	10110011	001001
pixel (0,3)	15060664	00011101	01110011	10100111	pixel (5,3)	14994871	11101101	10110011	001001
pixel (0,4)	14994871	11101101	10110011	00100111	pixel (5,4)	14929332	00101101	10110011	110001
pixel (0,5)	14863285	10101101	11010011	01000111	pixel (5,5)	14929332	00101101	10110011	110001
pixel (0,6)	15060405	10101101	10110011	10100111	pixel (5,6)	15126453	10101101	11110011	011001
pixel (0,7)	15126198	01101101	01110011	01100111	pixel (5,7)	14797488	00001101	01010011	100001
pixel (0,8)	14994867	11001101	10110011	00100111	pixel (5,8)	14797488	00001101	01010011	100001
pixel (0,9)	14994867	11001101	10110011	00100111	pixel (5,9)	15192759	11101101	01001011	111001
pixel (1,0)	14929587	11001101	01110011	11000111	pixel (6,0)	14863539	11001101	00110011	010001
pixel (1,1)	14863539	11001101	00110011	01000111	pixel (6,1)	14929332	00101101	10110011	110001
pixel (1,2)	14929078	01101101	00110011	11000111	pixel (6,2)	14995125	10101101	01110011	001001
pixel (1,3)	14994871	11101101	10110011	00100111	pixel (6,3)	14929332	00101101	10110011	110001
pixel (1,4)	14929078	01101101	00110011	11000111	pixel (6,4)	14863539	11001101	00110011	010001
pixel (1,5)	14797492	00101101	01010011	10000111	pixel (6,5)	14863539	11001101	00110011	010001
pixel (1,6)	15126198	01101101	01110011	01100111	pixel (6,6)	15192246	01101101	00001011	111001
pixel (1,7)	14928819	11001101	11010011	11000111	pixel (6,7)	14929074	01001101	00110011	110001
pixel (1,8)	14863281	10001101	11010011	01000111	pixel (6,8)	14732208	00001101	11010011	000001
pixel (1,9)	15060660	00101101	01110011	10100111	pixel (6,9)	15061173	10101101	00001011	101001
pixel (2,0)	14929332	00101101	10110011	11000111	pixel (7,0)	14863539	11001101	00110011	010001
pixel (2,1)	14863539	11001101	00110011	01000111	pixel (7,1)	14995125	10101101	01110011	001001
pixel (2,2)	14929078	01101101	00110011	11000111	pixel (7,2)	15060918	01101101	11110011	101001
pixel (2,3)	14994871	11101101	10110011	00100111	pixel (7,3)	14995125	10101101	01110011	001001
pixel (2,4)	14994871	11101101	10110011	00100111	pixel (7,4)	14929332	00101101	10110011	110001
pixel (2,5)	14929078	01101101	00110011	11000111	pixel (7,5)	14929332	00101101	10110011	110001
pixel (2,6)	15060405	10101101	10110011	10100111	pixel (7,6)	15060660	00101101	01110011	101001
pixel (2,7)	14928819	11001101	11010011	11000111	pixel (7,7)	15192759	11101101	01001011	111001
pixel (2,8)	14994867	11001101	10110011	00100111	pixel (7,8)	15061173	10101101	00001011	101001
pixel (2,9)	15126453	10101101	11110011	01100111	pixel (7,9)	14864308	00101101	11110011	010001
pixel (3,0)	14929332	00101101	10110011	11000111	pixel (8,0)	14797746	01001101	11010011	100001
pixel (3,1)	14929332	00101101	10110011	11000111	pixel (8,1)	14995125	10101101	01110011	001001
pixel (3,2)	14994871	11101101	10110011	00100111	pixel (8,2)	14863539	11001101	00110011	010001
pixel (3,3)	15126457	10011101	11110011	01100111	pixel (8,3)	14995125	10101101	01110011	001001
pixel (3,4)	15126457	10011101	11110011	01100111	pixel (8,4)	14995380	00101101	11110011	001001
pixel (3,5)	15126711	11101101	00001011	01100111	pixel (8,5)	15192759	11101101	01001011	111001
pixel (3,6)	14863026	01001101	01010011	01000111	pixel (8,6)	15061173	10101101	00001011	101001
pixel (3,7)	15060405	10101101	10110011	10100111	pixel (8,7)	15192759	11101101	01001011	111001
pixel (3,8)	15258039	11101101	10001011	00010111	pixel (8,8)	15324345	10011101	00101011	100101
pixel (3,9)	15192246	01101101	00001011	11100111	pixel (8,9)	14930101	10101101	00001011	110001
pixel (4,0)	14863539	11001101	00110011	01000111	pixel (9,0)	14863539	11001101	00110011	010001
pixel (4,1)	14929332	00101101	10110011	11000111	pixel (9,1)	14995125	10101101	01110011	001001
pixel (4,2)	14994871	11101101	10110011	00100111	pixel (9,2)	14863539	11001101	00110011	010001
pixel (4,3)	15060664	00011101	01110011	10100111	pixel (9,3)	14929332	00101101	10110011	110001
pixel (4,4)	15126711	11101101	00001011	01100111	pixel (9,4)	14929587	11001101	01110011	110001
pixel (4,5)	15126711	11101101	00001011	01100111	pixel (9,5)	15061173	10101101	00001011	101001
pixel (4,6)	14928819	11001101	11010011	11000111	pixel (9,6)	15061173	10101101	00001011	101001
pixel (4,7)	14994867	11001101	10110011	00100111	pixel (9,7)	14929587	11001101	01110011	110001
pixel (4,8)	15192246	01101101	00001011	11100111	pixel (9,8)	15127480	00011101	11001011	011001
pixel (4,9)	15258039	11101101	10001011	00010111	pixel (9,9)	14733237	10101101	11110011	000001

النتائج :يمكن توضيحها بالصور الآتية :



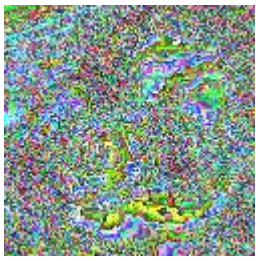
Laplacian operator

Cipher image
The key: 123456

Decipher image

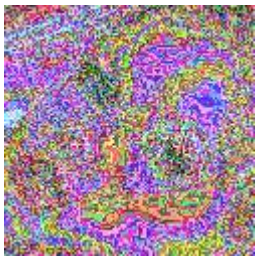
Cipher image
The key: 251463

Decipher image



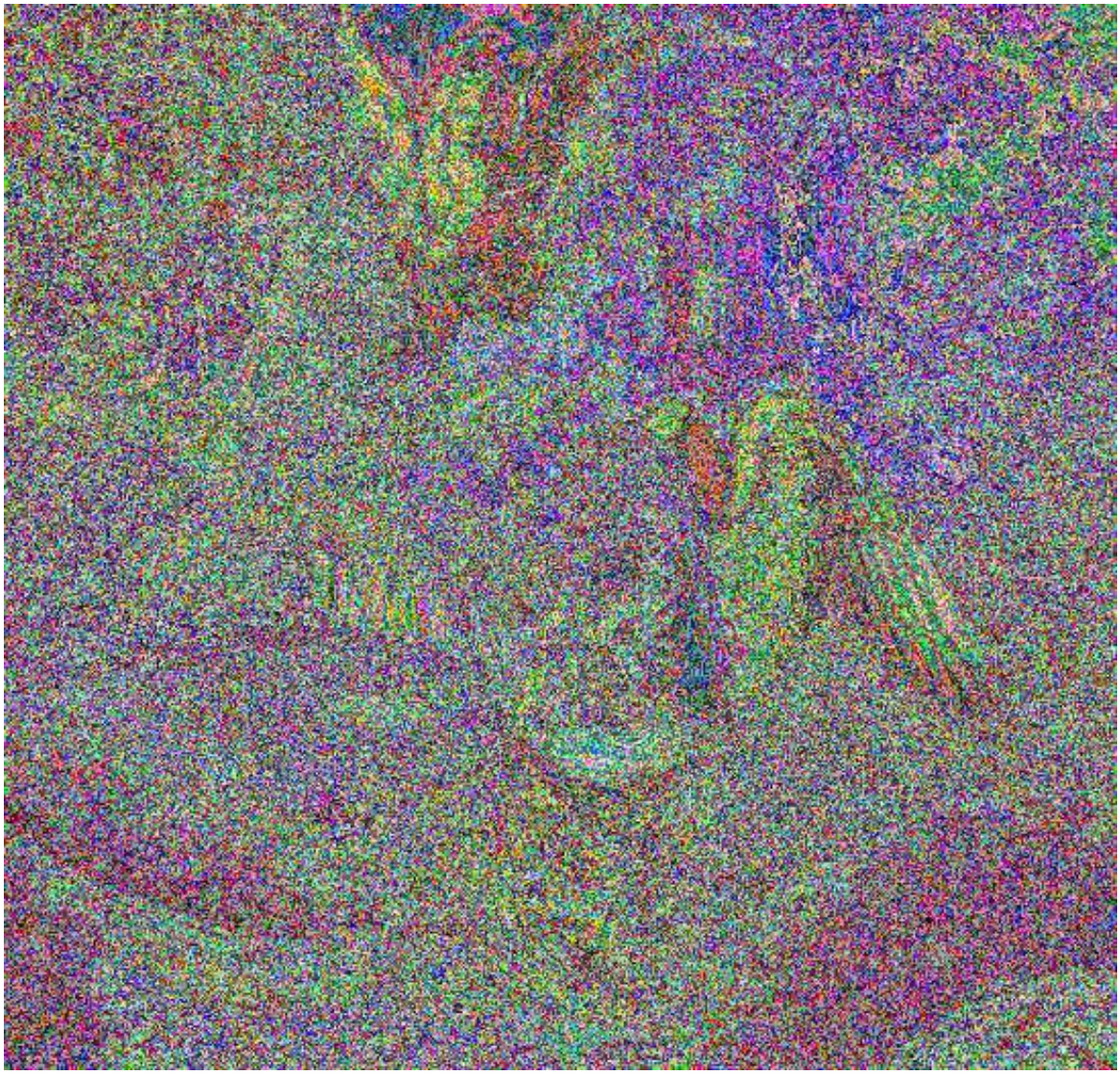
Cipher image
The key: 364251

Decipher image
The inverse of key:641352



Cipher image
The key: 241563

Decipher image
The inverse of key: 316245



Cipher image
The key: 321654

Decipher image
The inverse key: 321654

النتائج:

1. إن إجراء عملية التشفير لصورة معينة تعتمد اعتماداً كلياً على مقدار الدقة الموجودة في الصورة (نسبة الشـــدة اللونية في الصورة) وتحديد حواف اللون المؤثر فيها حيث كلما كانت الشدة اللونية ذات(دقة عالية) احتجنا إلى استخدام مفاتيح تشفير جديدة لحين الحصول على مفتاح مؤثر لإجراء عملية التشفير.
وبالعكس من ذلك كلما كانت الشدة اللونية قليلة قلت كثافة الحواف لألوان الصورة كانت عملية التشفير أسهل حيث ستقل نسبة اختيار مفتاح من مفاتيح التشفير الابدالية .
2. إن من بين (24) مفتاح الممكنة لإجراء عملية التشفير فانه هناك مفتاحاً سيكون من المفاتيح الموثرة, سيستخدم لإجراء عملية التشفير وتهمل الباقية .
3. إن المفتاح المستخدم في تشفير صورة معينة تشفيراً كاملاً (100 %) قد لا ينتج في تشفير صورة أخرى بنسبة (10%), وهذا الأمر يحتاج منا إجراء محاكاة للشدة اللونية لهذه الصورة حتى تتم عملية التشفير بنسبة (100%).
4. إن من الصعب جدا فك الشفرة الصورة والتعرف على معالمها حتى وان تم التعرف على حواف ألوان الصورة المؤثرة والغير مؤثرة.
5. إن الوقت المستغرق لإجراء عملية تشفير أو فك تشفير للصورة المطبقة تعتمد على حجم الصورة المستخدم بالدرجة الأساس. ومن ثم مواصفات الحاسب المستخدم من حيث السرعة والذاكرة ،حيث استغرقت عملية التشفير في حاسب بنتيوم 4 ,(GHz 1700) (RAM256) بضعة دقائق تتراوح بين (0.5-1 دقيقة) بالنسبة لصور تتراوح احجامها بين 125*125، إما في الصور الأكبر حجماً فقد استغرقت إلى أكثر من ذلك.
6. أثبتت الطريقة كفاءة (100%) في عملية تشفير الصورة الملونة ذات الامتداد BMP.

References

[1] John R .Smith and shih –Fu chang ," Tools and techniques for color image retrieval " , is&/spiel proceedings VOL. 2670 , storage &retrie Val for image and video databases IV , February 1 , 1996.
[2] Jain A.K., "Fundamentals of Digital Image processing", prentice hall,1998.
[3] T.Lourens,k .Nakadai ," Graph extraction from color images ", ESNN 2001proceeding dings European symposium on Artificial neural networks Bruges (Belgium),25-27 April 2001 , D-Facto public,.
[4] Marshall F.Tappen , Bryan C .Russsell , williamt . Freeman ," Efficient Graphicall Models for processing images ", DRAFT, CVPR 2004, Cambridge.
[5] Ronan Fablet , pathrck Bouthemy , Marc Gelgon , " Moving Object detection in color Image sequences using region –level graph labeling ",proc. 6th IEEE int , cof.on Image , processing , ICIP'99 , KOBe, October, 1999.
[6] Scohh E.Umbaugh, " computer vision and image processing" ,prentice hall,1998.