Visual Cryptography Kit

THE OLIVETTI & ORACLE RESEARCH LABORATORY

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

http://www.cl.cam.ac.uk/~fms27/vck/

Frank Stajano 1998

Visual cryptography (invented by Naor & Shamir in 1994) is a method for securely encrypting messages in such a way that the recipient won't need a computer to decrypt them.
The underlying cipher is essentially the one time pad; so the system is unbreakable in the information theoretical sense.

The Visual Cryptography Kit, freely downloadable from the URL above, is a Python module, based on PIL and Tkinter, that allows easy practical experimentation with this fascinating invention.

One time pad

```
plaintext  = 010010101010111101001110010...
      To encrypt: xor
key        = 110011010010101000010101000...
                   =
ciphertext = 100001111000010101011011010...
                        To decrypt: xor
key        = 110011010010101000010101000...
                                     =
decrypted  = 010010101010111101001110010...
```

The mechanism of the one time pad is very simple: XOR every plaintext bit with the corresponding random key bit to yield a ciphertext bit: $c = p \oplus k$.
To decrypt, XOR the ciphertext with the key once more: $d = c \oplus k$. The two keys will cancel out and you'll get the plaintext.
$d = c \oplus k = (p \oplus k) \oplus k = p \oplus (k \oplus k) = p \oplus 0 = p$.

The one time pad is the only demonstrably secure cipher, in the sense that even an infinite amount of ciphertext will not leak any bits of information to the attacker about the plaintext (except the length). This is because, given a ciphertext, you can choose any plaintext you want and there will always exist a key that generates that ciphertext from that plaintext.
Why do people ever use anything else, then? Well, the one time pad is not very practical because it requires a lot of key material (as many key bits as message bits). You can't ever reuse the same key bits and you must use a truly random source; otherwise, cryptanalitic attacks become possible.
If you use a pseudo-random number generator, you instead obtain what is known as a stream cipher.

Pixelcoding – how to make XOR out of OR

The idea of visual cryptography is to perform a visual one time pad by overlaying transparent acetate sheets. But overlaying corresponds to OR, not to XOR: ink overlay ink gives ink, not transparent.
This is where the clever part of Naor & Shamir's idea kicks in: a new encoding convention for the pixels (different for the input pixels and the output ones) that allows XOR to be built out of OR plus thresholding; and the thresholding can be done "for free"by our visual system!

At first it appears obvious that overlay will never work, because one can't make $1 \oplus 1 = 0$ with it.

or	0	1
0	0	1
1	1	1

overlay	□	■
□	□	■
■	■	■

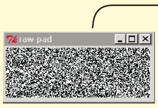
xor	0	1
0	0	1
1	1	0

overlay	◧	◨
◧	◧	■
◨	■	◨

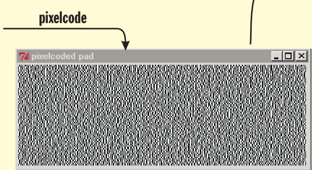
But try this: use ◧ for 0 and ◨ for 1 at the input; and accept both ◧ and ◨ as 0, and ■ for 1, at the output of the operation. Then note how
◨ overlay ◨ = ◨ i.e. $1 \oplus 1 = 0$ …
This clever trick, which I call pixelcoding, is the fundamental intuition of Naor & Shamir's visual cryptography construction.

On output from the decrypting XOR operation, a black pixel means black and a 50% grey pixel means white. This is a reduction in contrast, but our eyes will easily "see" the grey as white for free.
Note that on the input to the XOR, both black and white are represented as 50% grey — but by two complementary greys, i.e. one grey pattern has black where the other has white and vice versa. So the overlap of two identical input patterns yields the same pattern (a 50% grey, so logical white for output), while two opposite input patterns yield black everywhere. This satisfies the truth table for XOR.

Enciphering a bitmap image



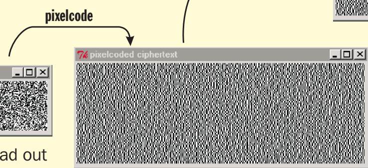
M chooses a size (width x height) for the message and generates a random pad of this size, made of black and white pixels. He stores it in his safe.



M pixelcodes the pad and gives the result to 007 on acetate. 007 goes to the Bahamas and has an active sex life as usual.

A couple of weeks later, M must tell 007 something really secret. He writes it on an image the size of the pad.

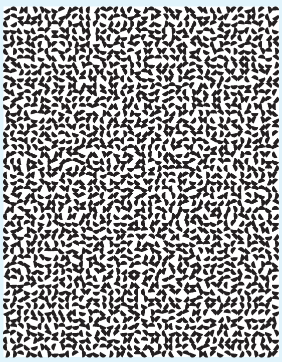
M gets the pad out of the safe and XORs it with the message. He then burns the pad.



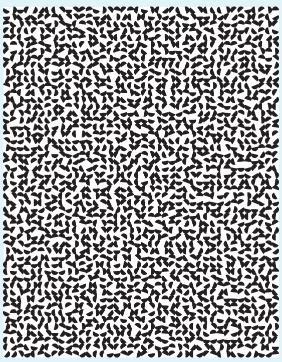
M pixelcodes the outcome of the XOR, obtaining a ciphertext. He faxes this to 007. SPECTRE taps the fax line, but is none the wiser. Ha ha ha!

Between a "shaken, not stirred" and a romantic dinner, 007 gets the fax. He overlays his acetate on it and reads M's message. (No computers needed!) He then burns both the fax and the acetate and proceeds to save the world.

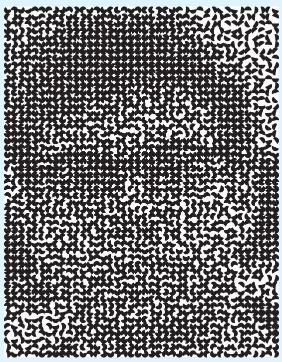
A variant for greyscale images



Halfmoons in the pad are randomly oriented.



Halfmoons in the ciphertext are oriented so that, when superimposed on the pad…



…they create pie angles proportional to the pixel intensities of the plaintext.

This artificial composite shows each halfmoon pair on a local background with the intensity of the original plaintext pixel: the fully open pairs are on white, the fully closed ones on black.

In practice, though, things never go as smoothly as this, neither for the monochrome nor for the greyscale variants. This is because, apart from the interaction between adjacent pixel groups, it is quite difficult to achieve proper registration over the surface of the whole picture. This is primarily due to the fact that the acetates expand in a non-uniform and irregular way during the printing process.