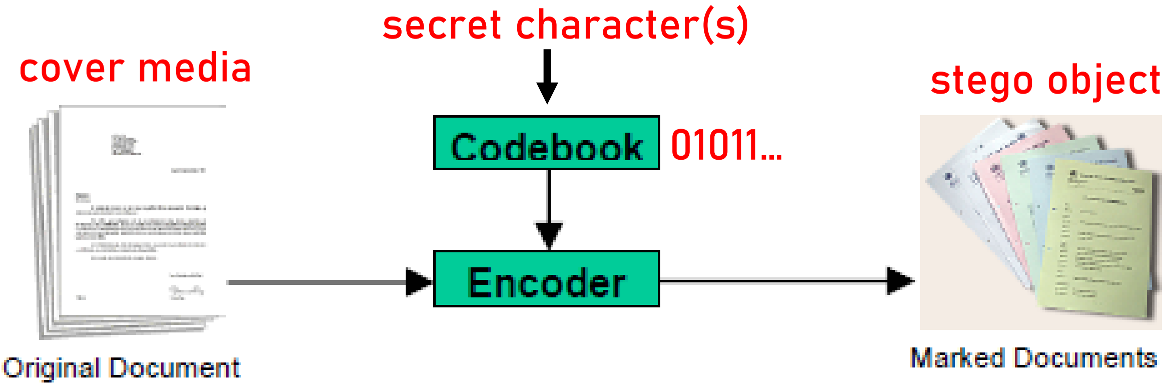
Plaintext Steganography (cont.)

1.2) Plaintext Steganography: Line Shifting or Word Shifting in Cover Media

- e.g., lines are shifted down by a small fraction
  - ➢ shift present = 1, shift not present = 0
- e.g., words are shifted right by a small fraction
  - ➢ shift present = 1, shift not present = 0
- encoded bits are extracted and compared against a predefined Codebook



cover media

secret character(s)

Codebook

01011...

Encoder

stego object

Original Document

Marked Documents

https://www.cs.bham.ac.uk/~mdr/teaching/modules03/security/students/SS5/Steganography.htm

Plaintext Steganography (cont.)

Example: Steganography with Line Shifting

Recipient's perspective!!! (message extraction)

IF you can keep your head when all about you
Are losing theirs and blaming it on you,

If you can trust yourself when all men doubt you,
But make allowance for their doubting too;
If you can wait and not be tired by waiting,
Or being lied about, don't deal in lies,
Or being hated, don't give way to hating,
And yet don't look too good, nor talk too wise:

If you can dream - and not make dreams your master;
If you can think - and not make thoughts your aim;
If you can meet with Triumph and Disaster

And treat those two impostors just the same;
If you can bear to hear the truth you've spoken
Twisted by knaves to make a trap for fools,
Or watch the things you gave your life to, broken,
And stoop and build 'em up with worn-out tools:

If you can make one heap of all your winnings
And risk it on one turn of pitch-and-toss,
And lose, and start again at your beginnings
And never breathe a word about your loss;
If you can force your heart and nerve and sinew

To serve your turn long after they are gone,
And so hold on when there is nothing in you
Except the Will which says to them: 'Hold on!'

If you can talk with crowds and keep your virtue,

Or walk with Kings - nor lose the common touch,
If neither foes nor loving friends can hurt you,
If all men count with you, but none too much;
If you can fill the unforgiving minute
With sixty seconds' worth of distance run,
Yours is the Earth and everything that's in it,
And - which is more - you'll be a Man, my son!

extract

Codebook:

00000 – a	10010 – n
00001 – b	10011 – o
00010 – c	10100 – p
00011 – d	10101 – q
00100 – e	10110 – r
00101 – f	10111 – s
00111 – g	11000 – t
01000 – h	11001 – u
01001 – i	11010 – v
01011 – j	11011 – w
01111 – k	11100 – x
10000 – l	11101 – y
10001 – m	11110 – z

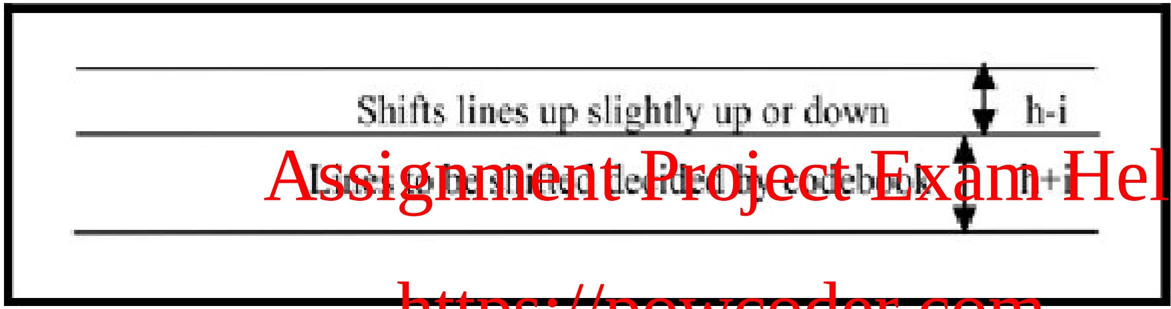
01000
00100
10000
10000
10011

decode

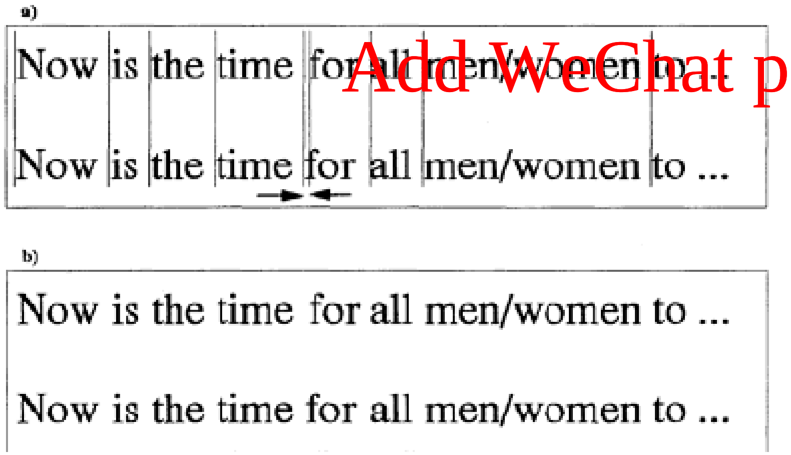
hello

Plaintext Steganography (cont.)

Example: More Subtle Forms of Line and Word Shifting


Shifts lines up slightly up or down
h-i
Lines to be shifted decided by codebook
h+i

Text placed at the bottom vs. at the center of a row.


a)
Now is the time for all men/women to ...
Now is the time for all men/women to ...
b)
Now is the time for all men/women to ...
Now is the time for all men/women to ...

https://www.researchgate.net/publication/228672143_WhiteSteg_A_new_scheme_in_information_hiding_using_text_steganography/figures?lo=1

https://www.researchgate.net/publication/228672143_WhiteSteg_A_new_scheme_in_information_hiding_using_text_steganography/figures?lo=1

Image Steganography

In each point, there could be an infinite number of possible color shades.
No two (adjacent) points could be of the same color.

Bonus Question:

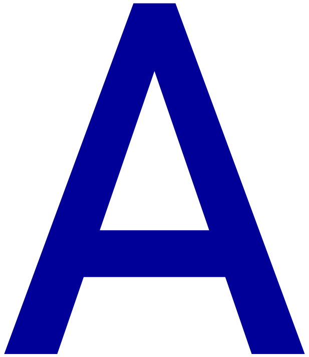
Is it possible to 'faithfully' store a classical painting in a digital form? Explain!

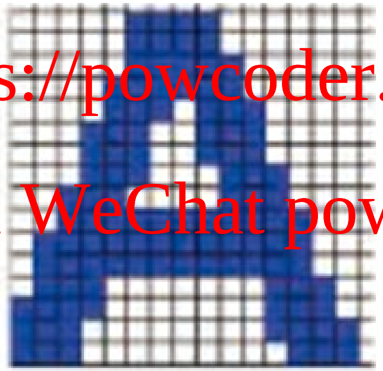
Image Steganography

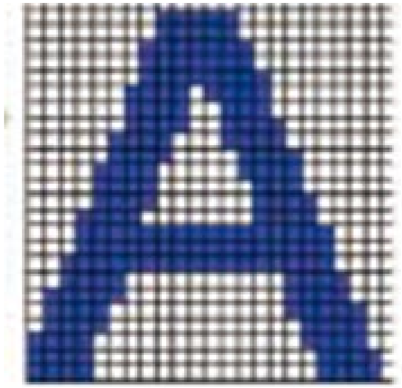
Image is broken into a finite number of areas that contain the same color/shade.
There is finite number of colors/shades available.

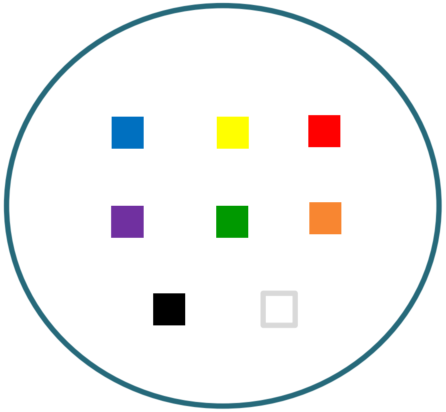
Example: Digitized Image

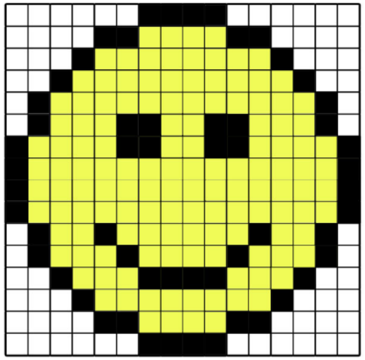
- any image can be digitized – i.e., represented by a discrete set of **same-color points / display elements**









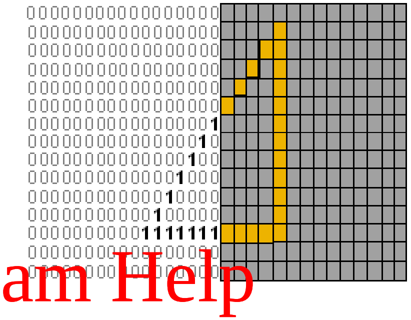


How can we improve the quality of digital images?

By decreasing the size of display elements (increase resolution).

Image Steganography (cont.)

- **Digital Image** –
  a 2D (NxM) array/grid
  of m-bit pixels



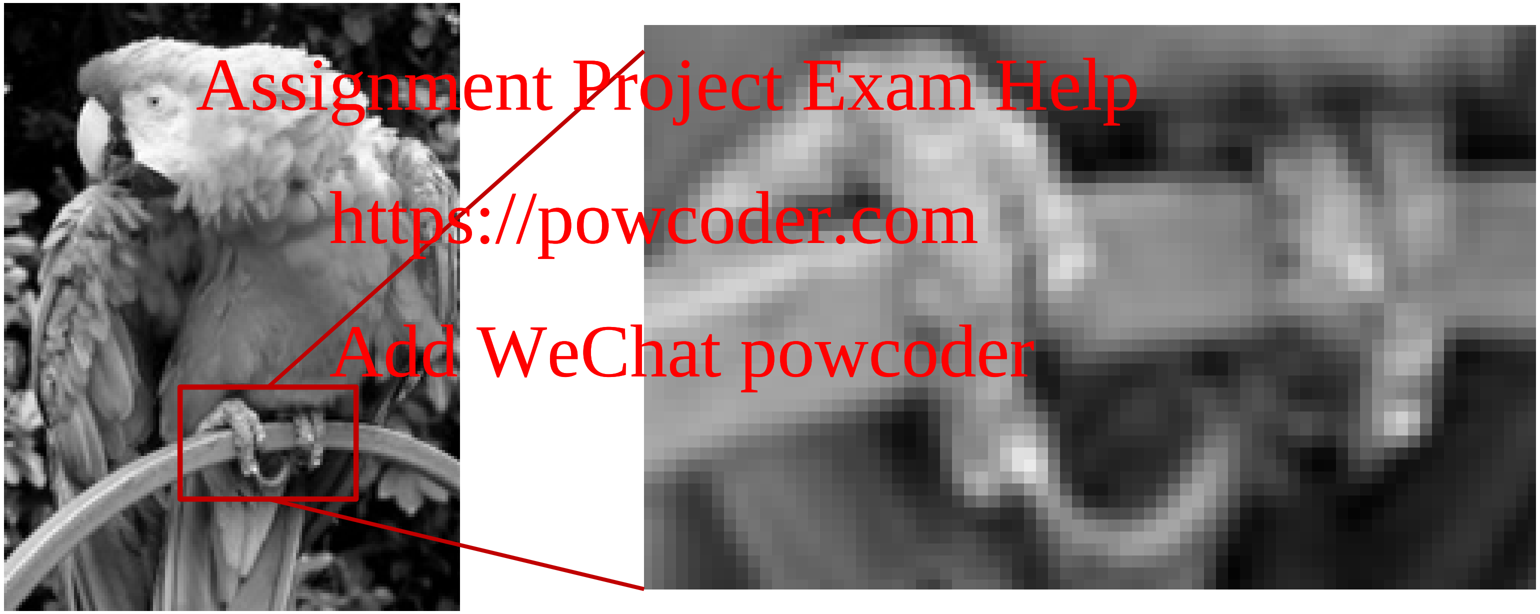
```
00000000000000000
00000000000000000
00000000000000000
00000000000000000
00000000000000000
00000000000000000
00000000000000001
00000000000000010
00000000000000100
00000000000001000
00000000000010000
00000000000100000
00000000001111111
00000000000000000
00000000000000000
```

- **Pixel** – fundamental same-color display element in a digital image

  - each pixel is made up of one or more bits
    - monochrome image: pixel = 1 bit => (black/white)
    - grayscale image: pixel = 8 bits => 256 shades of gray
    - RGB image: pixel = 24 bits => 8 bits for each – red, green, blue => 16777216 different color shades

Image Steganography (cont.)

Example: Pixels in Grayscale Image



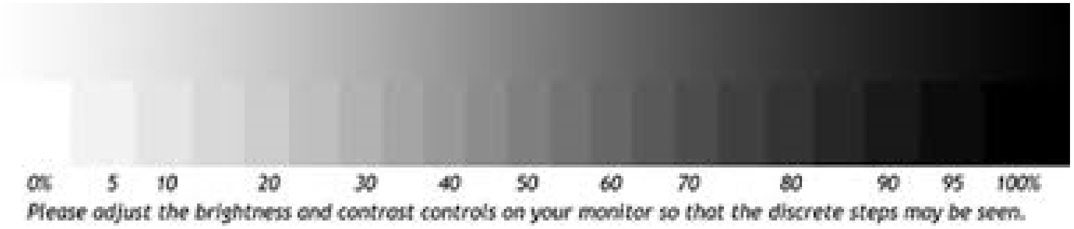
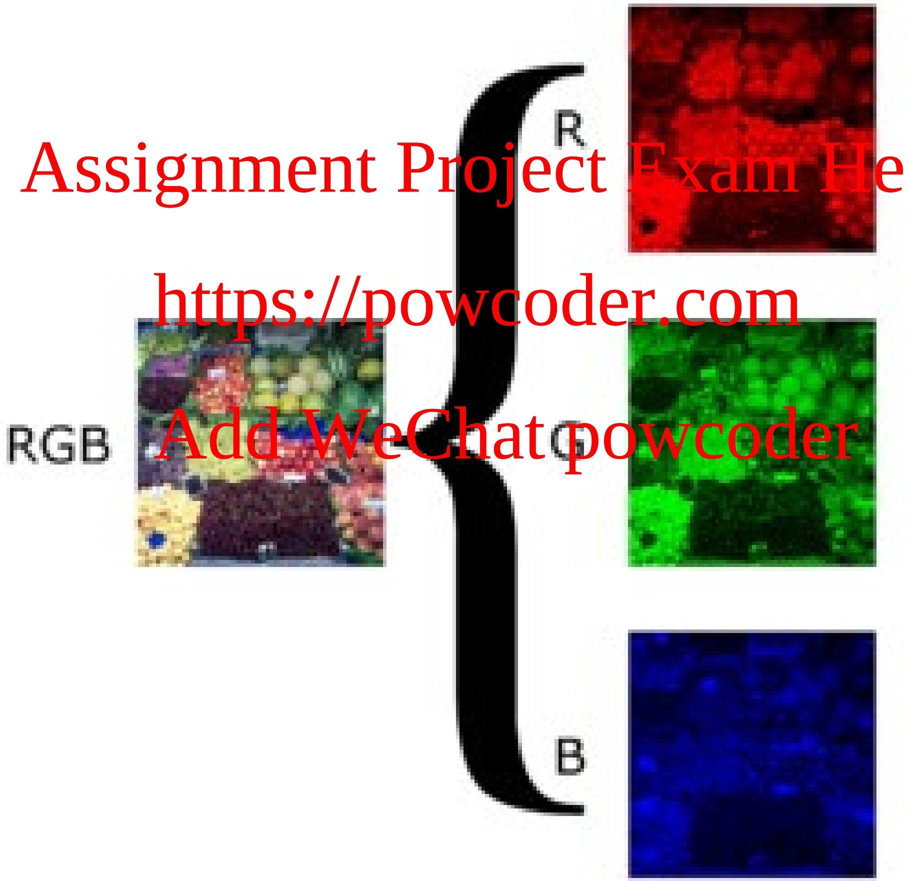

0%
5
10
20
30
40
50
60
70
80
90
95
100%
Please adjust the brightness and contrast controls on your monitor so that the discrete steps may be seen.

Image Steganography (cont.)

Example: RGB Image



http://www.freeonlinephotoeditor.com/

Image Steganography (cont.)

<u>Example</u>: Image Size

What is the size (in kbytes and KBytes) of a <u>grayscale image</u> comprising 200x300 pixels?

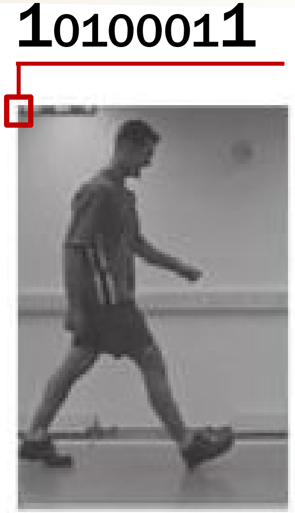
200 x 300 x 8 = 480,000 bits
= 60,000 bytes
= 60 kbytes
= 58.59 KBytes

kbyte = 10^3 bytes = 1000 bytes
KByte = 2^{10} bytes = 1024 bytes

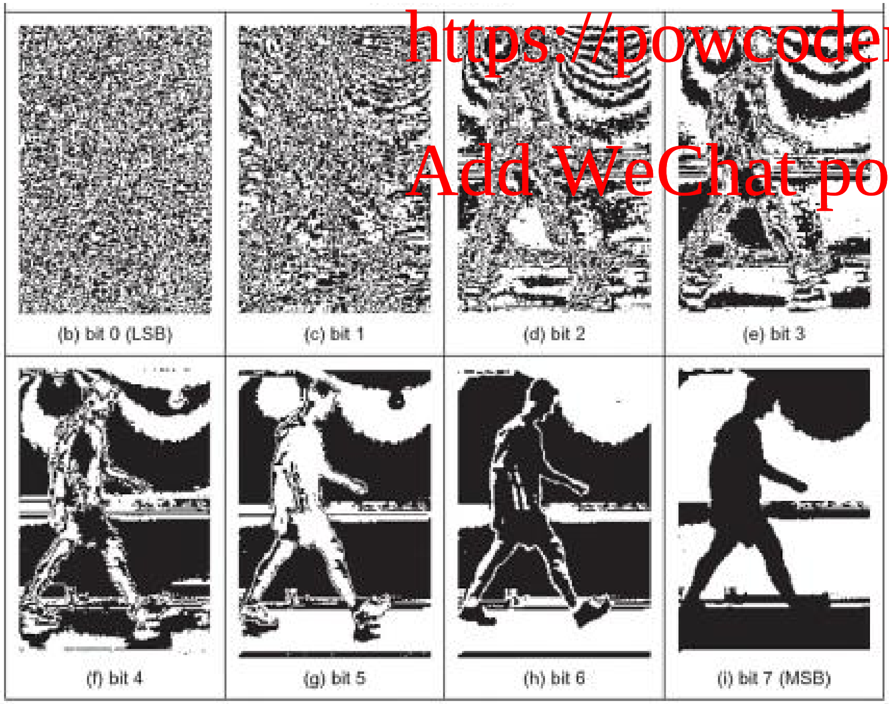
http://jan.ucc.nau.edu/lrm22/pixels2bytes/calculator.htm

Image Steganography (cont.)

- **Bits in a Pixel** – relative importance of different pixels is different
  - LSB – least significant bit – last bit
  - MSB – most significant bit – 1^{st} bit



(a) Original image



(b) bit 0 (LSB) (c) bit 1 (d) bit 2 (e) bit 3

(f) bit 4 (g) bit 5 (h) bit 6 (i) bit 7 (MSB)

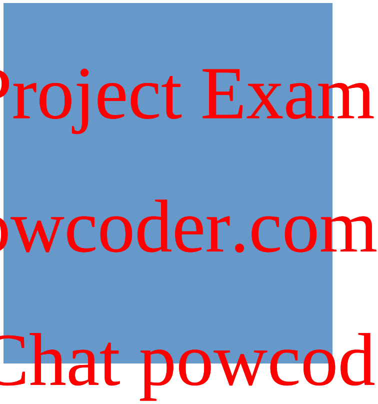
FIGURE 2.1

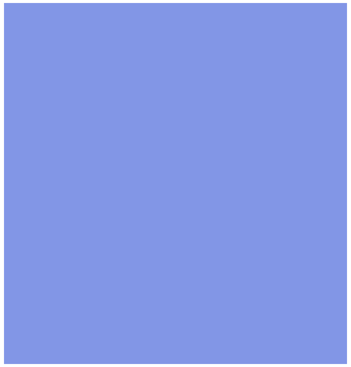
Decomposing an image into its bits.

- LSB carries the least information – it changes most rapidly
- MSB carries the most information – it changes least rapidly

Image Steganography (cont.)

Example: LSB(s) and Human Eye





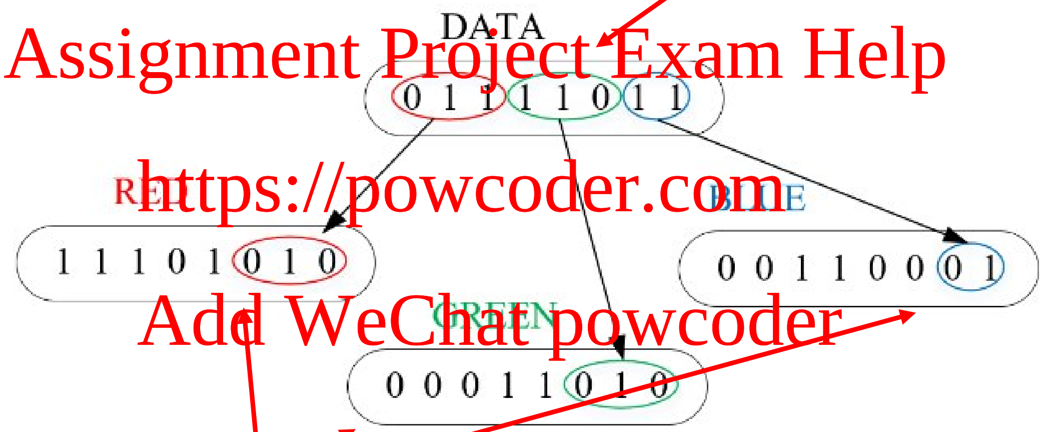
R = 100_{10} = 01100100	R = 102_{10} = 01100110	R = 130_{10} = 01111000
G = 150_{10} = 10010110	G = 152_{10} = 10011000	G = 150_{10} = 10010110
B = 200_{10} = 11001000	B = 202_{10} = 11001010	B = 230_{10} = 11100110

http://jumk.de/color-calculator/

Image Steganography (cont.)

Example: ‘text in image’ using LSB

This is a secret message.



cover image

https://www.researchgate.net/publication/321183782_FPGA_implementation_of_LSB-based_steganography/figures?lo=1

Image Steganography (cont.)

2.1) Image Steganography:

Use of LSB to hide '<u>image in image</u>'

- <u>easiest</u> and <u>surprisingly effective</u> way of hiding information in an image

- LSB(s) of each pixel is/are used to hide the most significant bits of another image

- algorithm:

(1) load up host image and image to hide

(2) choose the number of LSBs you whish to hide the secret image in

more bits used { => better quality of hidden image ☺
=> more distortion in cover image ☹ }

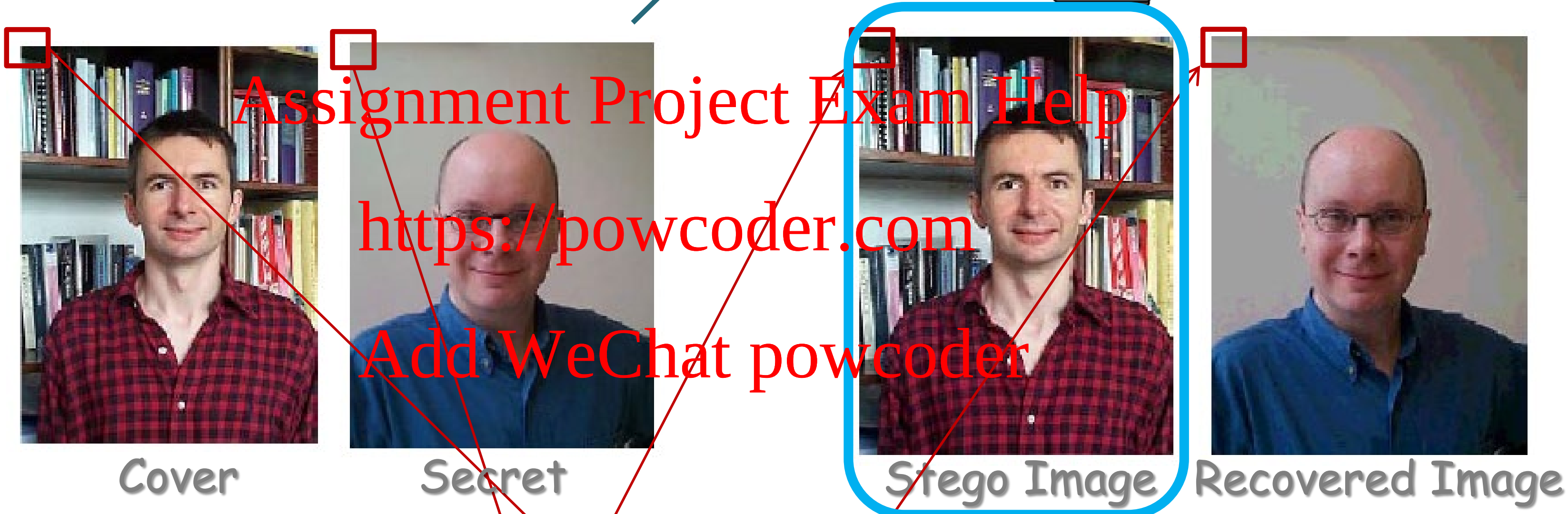
(3) to get original image back, pick out the LSBs according to the number used in (2)

Image Steganography (

Cover and secret image of the same size (# of pixels).
One secret pixel has to be 'encoded' in one cover pixel.

Example: '*image in image*' using LSB

Number of LSB used = 4



Encoding:

```
    Host Pixel: 10110001
  Secret Pixel: 00111111
New Image Pixel: 10110011
```

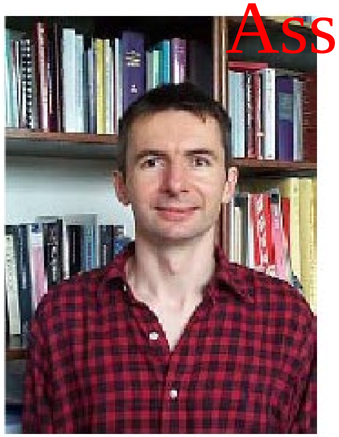
Decoding:

```
Host Pixel: 10110011
 Bits used: 4
 New Image: 00110000
```

Image Steganography (cont.)

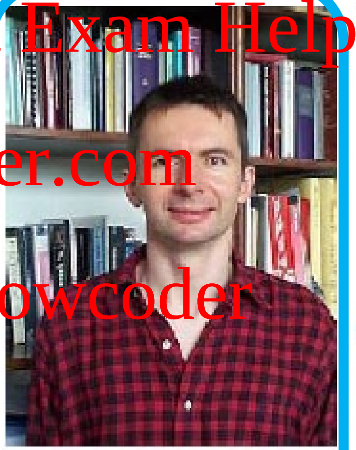
Example: 'image in image' using LSB

Number of LSB used = 1

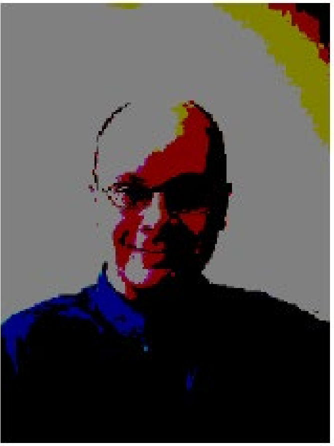


Cover

Secret



Stego Image



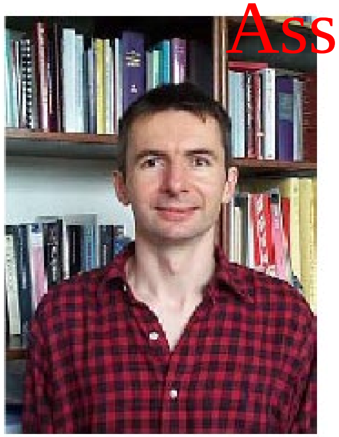
Recovered Image

fewer LSB bits used => 'hiding' capacity low –
better stego-image ☺
worse recovered image ☹

Image Steganography (cont.)

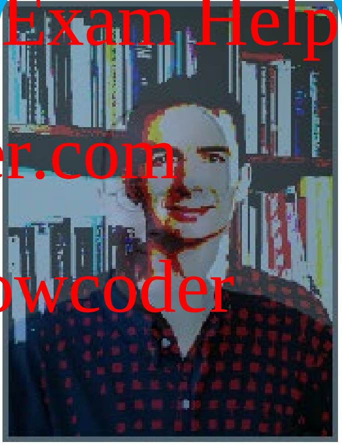
Example: ’image in image’ using LSB

Number of LSB used = 7

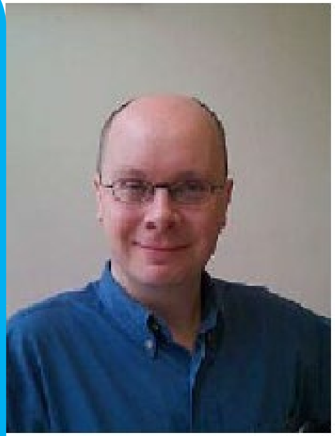


Cover

Secret



Stego Image

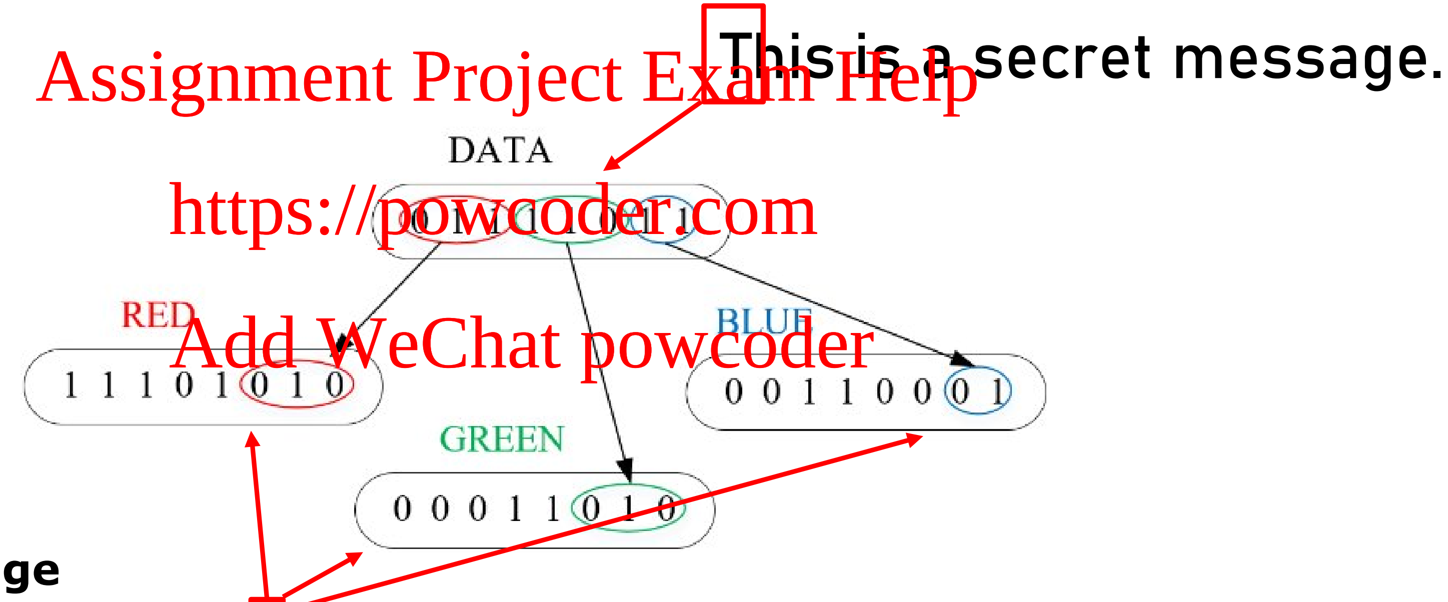


Recovered Image

more LSB bits used => ‘hiding’ capacity high –
worse stego-image ☹
better recovered-image ☺

Image Steganography (cont.)

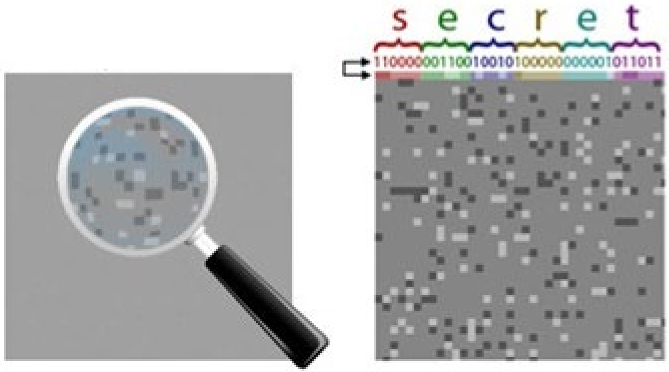
What if we do not need all the pixels of the cover image to hide our secret message??



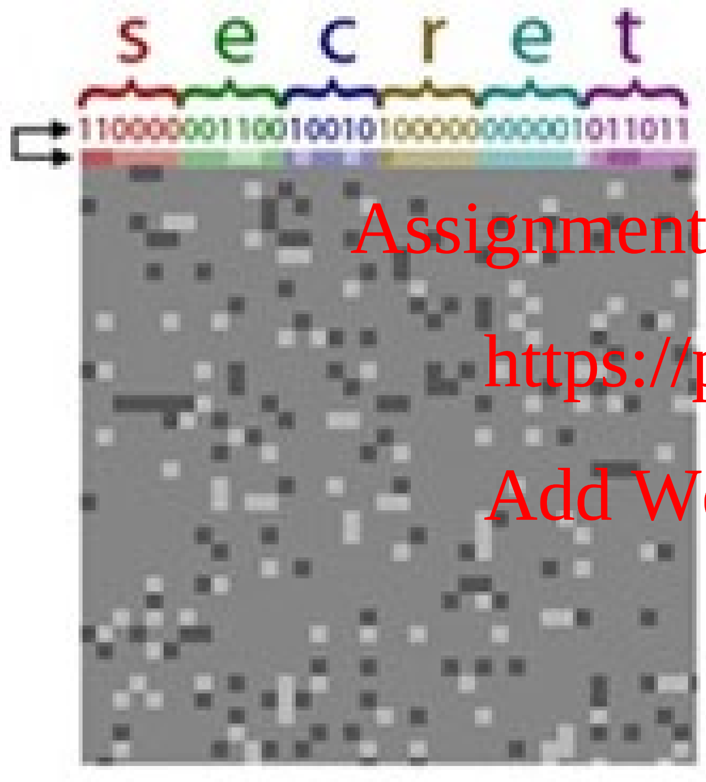
cover image

Image Steganography (cont.)

- **Pattern of LSB Embedding** – secret bits can be embedded in LSBs of cover image in two ways:
  - **sequentially**
    - simple embedding & extraction of secret bits ☺
    - statistics of cover image abruptly changed - easy to detect ☹
  - **randomly**
    - the key to generate pseudorandom numbers must be sent ☹
    - secret bits scattered throughout cover image - hard to detect ☺



Is ‘random’ choice of pixels an ideal approach to information hiding in an image ???



Should not ‘mess up’ pixel values in areas of ‘low entropy’.

What is a better place to hide secret bits:

- same-color background
- part of image with lots of detail ???

Image Steganography (cont.)

2.2) Image Steganography:
Use of Discrete Cosine Transform (DCT)

- DCT is one of key components of JPEG compression

- JPEG algorithm:

(1) algorithm is split in 8x8 pixel squares

(2) each square is transformed via DCT to 64 frequency components

(3) each DCT coefficient is quantized against a reference table – many bits get removed

- more bits are used for low-freq. and fewer for high-freq. components
  (human eye is more sensitive to low-freq. info)

(4) many coefficients are (now) close in value => run/variable length coding can be used

Image Steganography (cont.)

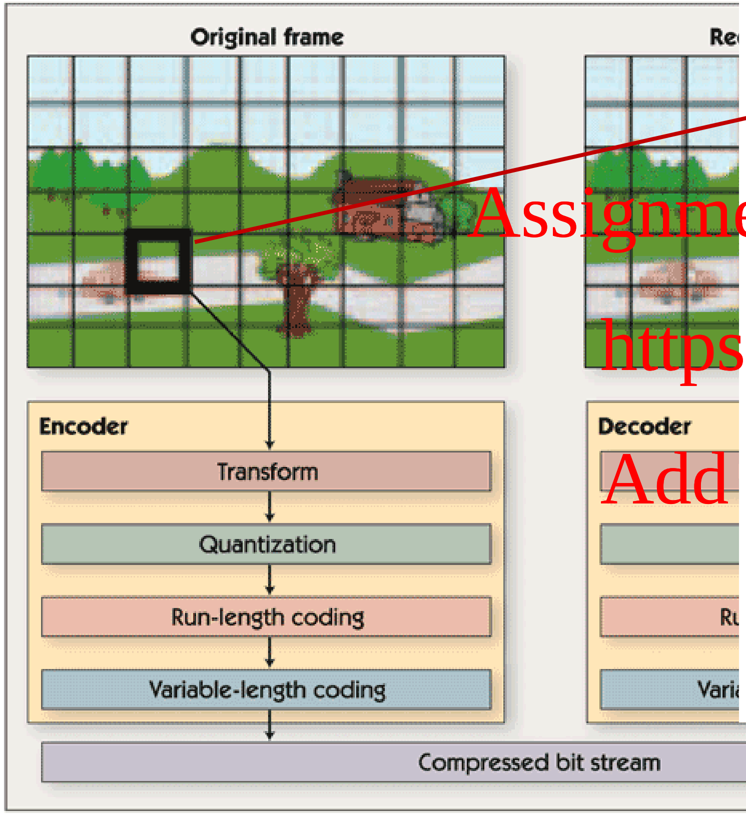


Fig. 1 Still image compression begins by dividing the image into 8-pixel by 8-pixel blocks. The main processing steps that follow are transformation to the frequency domain, quantization of the frequency domain coefficients, run-length coding of the coefficients, and variable-length coding. Still image decompression reverses these steps: variable-length decoding, run-length decoding, and dequantization restore the frequency domain coefficients (with some quantization error) and an inverse transform reconstructs the pixels in each image block from those coefficients.

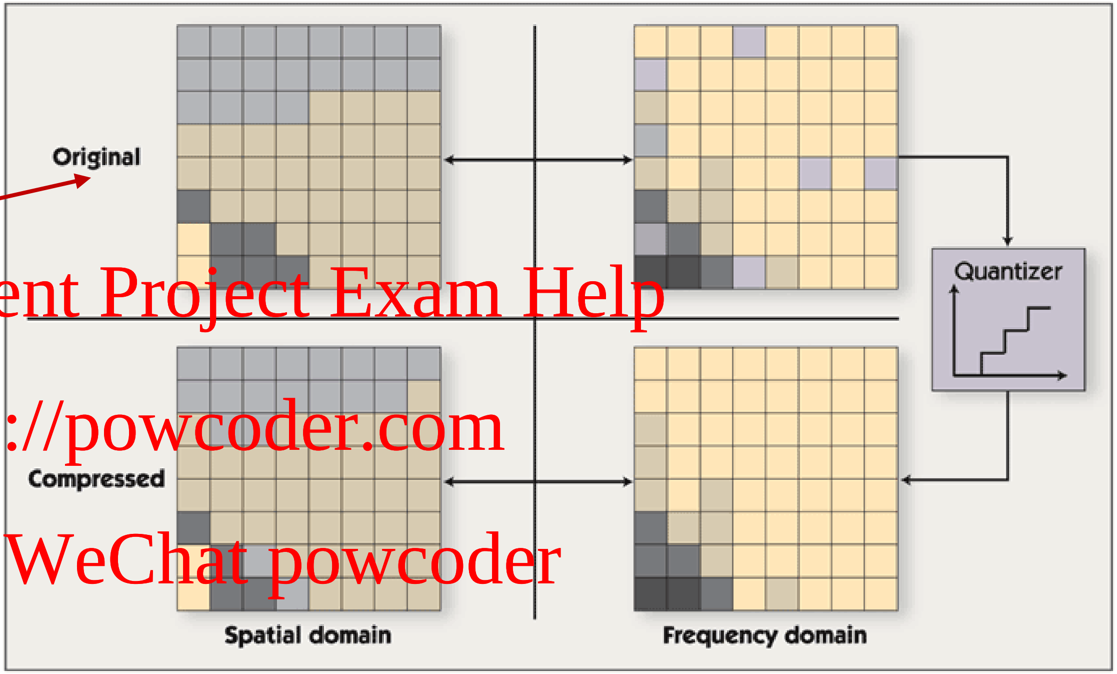


Fig. 3 Each frequency-domain coefficient is quantized in the encoder by rounding to the nearest of a number of pre-determined values. Because this rounding discards some information, the 8x8 block of pixels reconstructed by the decoder is close—but not identical—to the corresponding block in the original image.

http://www.eetimes.com/document.asp?doc_id=1275437

Image Steganography (cont.)

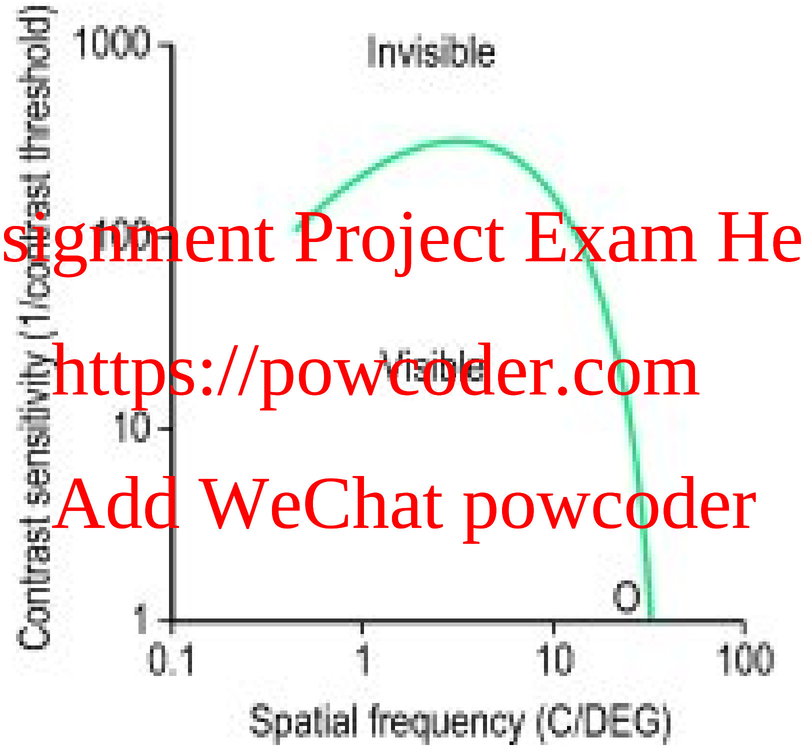
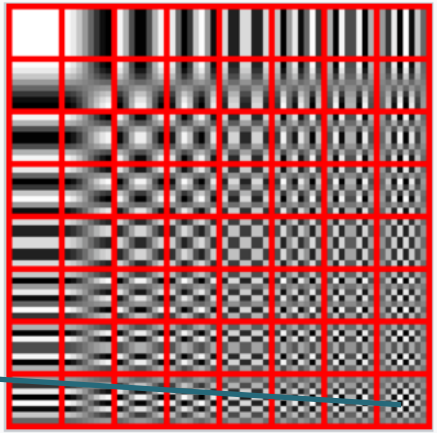


Fig. F10 Typical contrast sensitivity function of an adult human eye (O, cut-off frequency) (both scales are logarithmic)

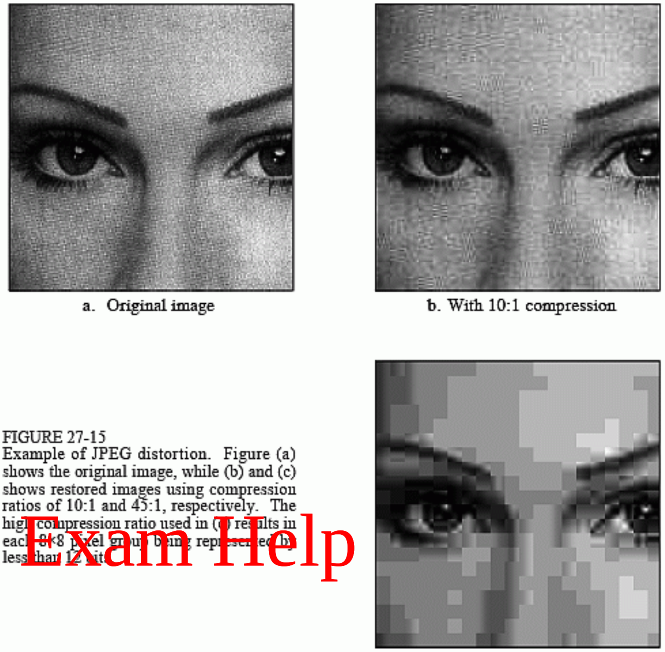
http://medical-dictionary.thefreedictionary.com/Function+(disambiguation)

Low Frequency (intensity)

High Frequency (intensity)



The DCT transforms an 8×8 block of input values to a linear combination of these 64 patterns. The patterns are referred to as the two-dimensional DCT *basis functions*, and the output values are referred to as *transform coefficients*. The horizontal index is u and the vertical index is v.



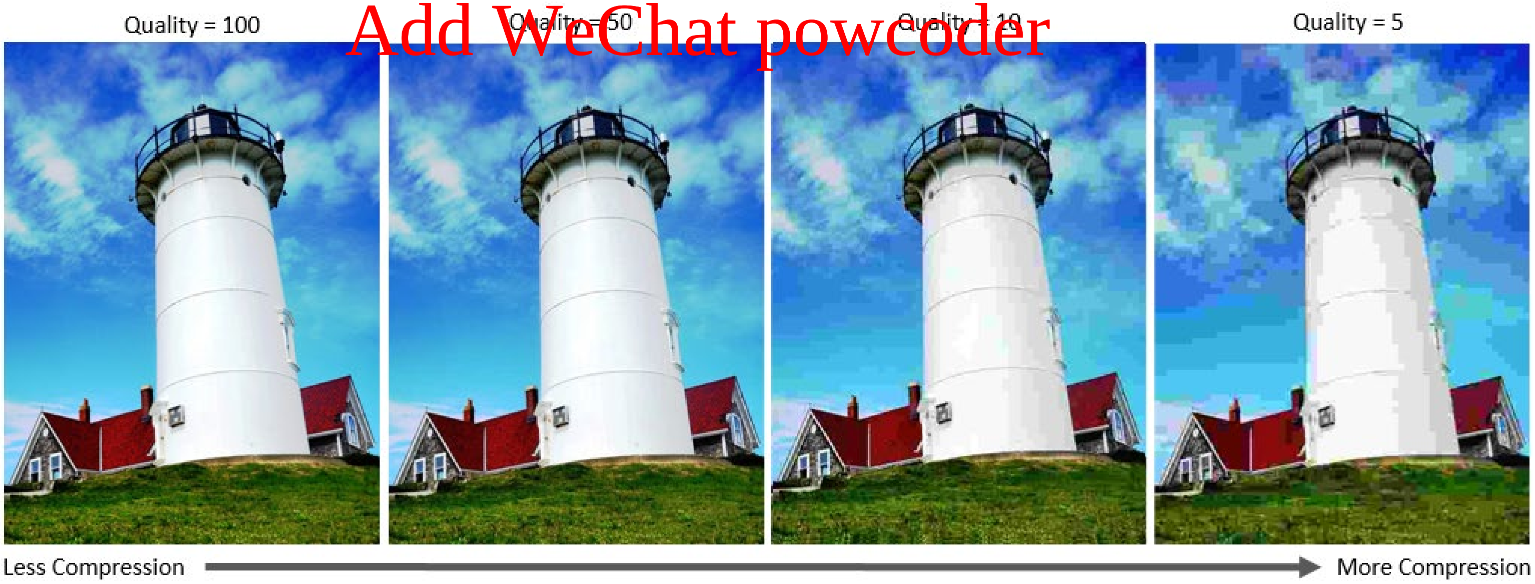
a. Original image

b. With 10:1 compression

c. With 45:1 compression

FIGURE 27-15
Example of JPEG distortion. Figure (a) shows the original image, while (b) and (c) shows restored images using compression ratios of 10:1 and 45:1, respectively. The high compression ratio used in (c) results in each 8×8 pixel group being represented by less than 12 bits.

https://www.dspguide.com/ch27/6.htm



https://www.mathworks.com/help/images/jpeg-image-deblocking-using-deep-learning.html

Image Steganography (cont.)

2.2) Image Steganography:
Use of Discrete Cosine Transform (DCT) - cont.

- Possible Approaches to Hiding Data in DCT

(A) hide secret data in LSBs of DCT coefficients

(B) hide secret data in LSBs of selected or non-significant DCT coefficients (frequencies)

(B) hide one bit of data in each 8x8 block of DCT:
0 => all coefficients even
1 => all coefficients odd

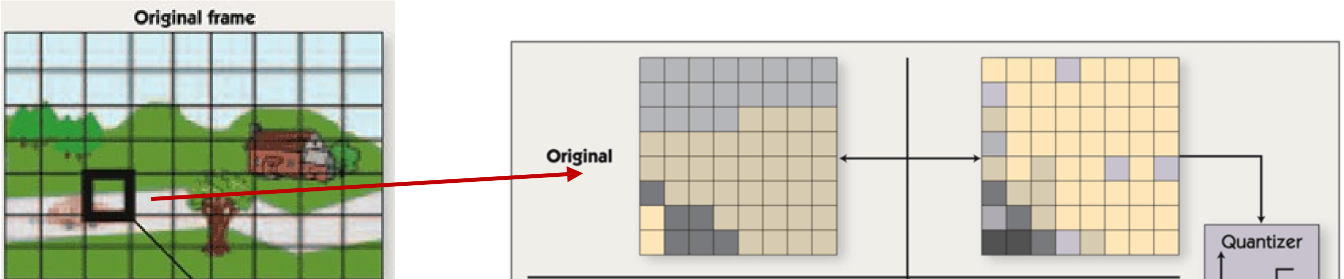
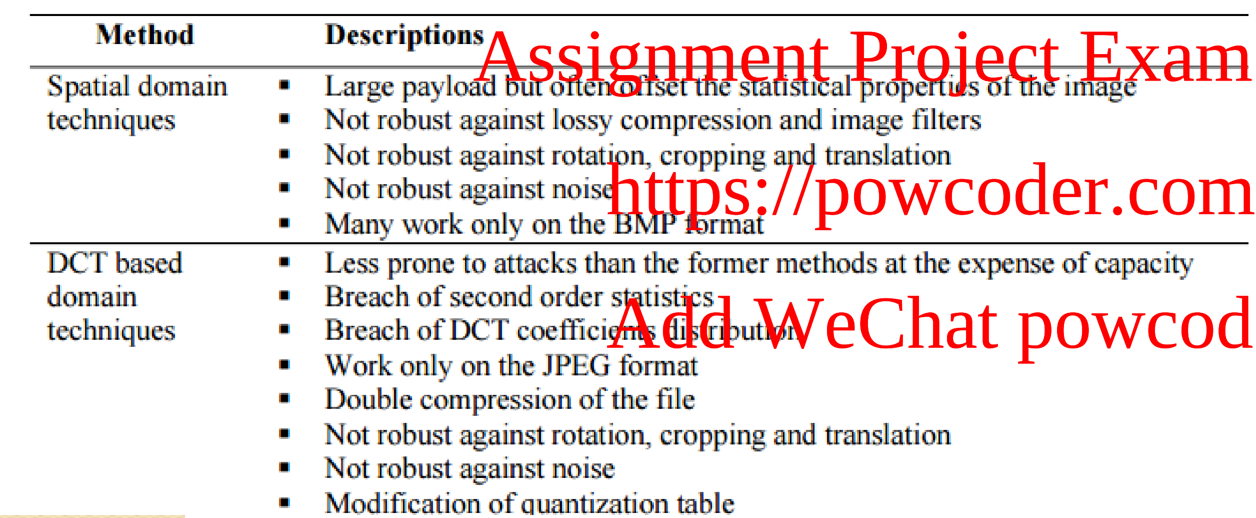
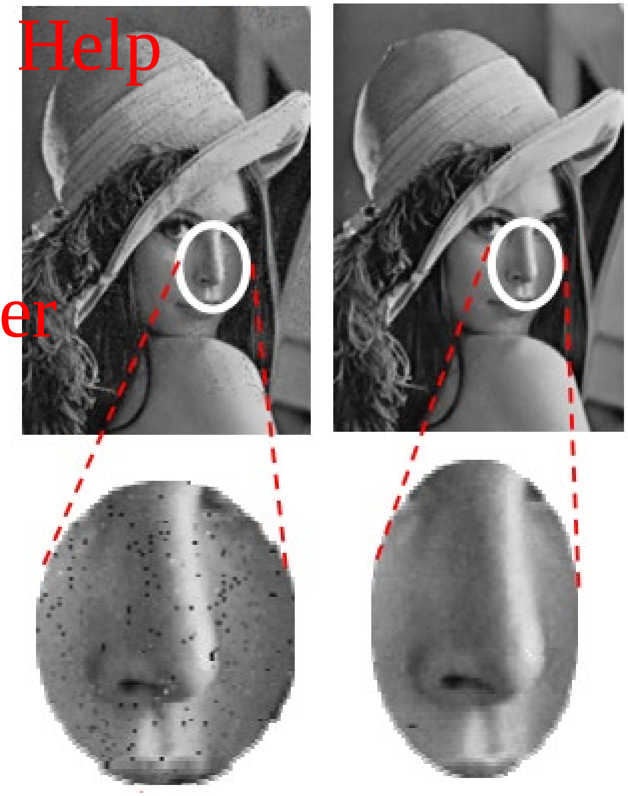


Image Steganography (cont.)

Example: Comparison of Spatial vs. DCT Based Steganography



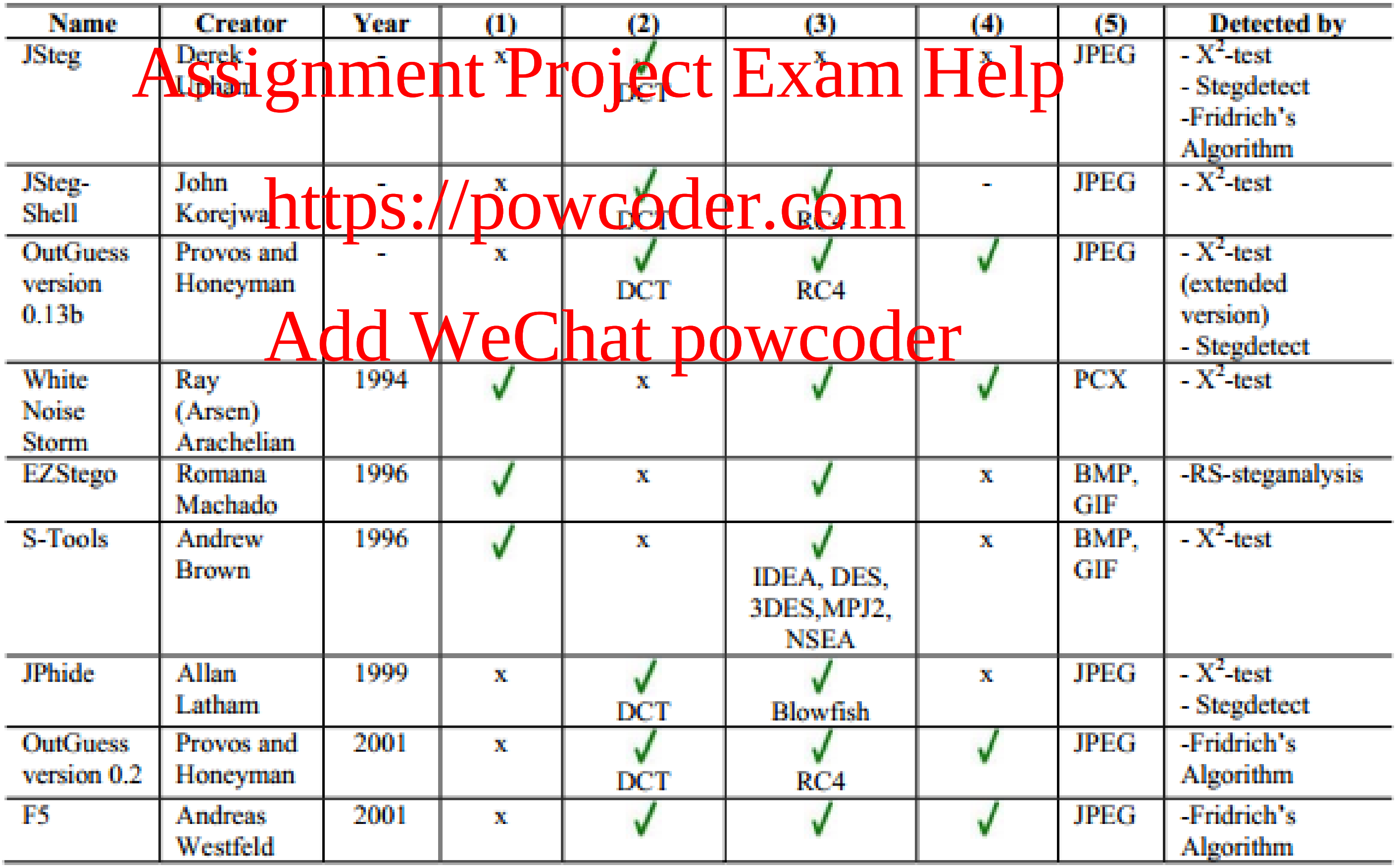
Method	Descriptions
Spatial domain techniques	▪ Large payload but often offset the statistical properties of the image ▪ Not robust against lossy compression and image filters ▪ Not robust against rotation, cropping and translation ▪ Not robust against noise ▪ Many work only on the BMP format
DCT based domain techniques	▪ Less prone to attacks than the former methods at the expense of capacity ▪ Breach of second order statistics ▪ Breach of DCT coefficients distribution ▪ Work only on the JPEG format ▪ Double compression of the file ▪ Not robust against rotation, cropping and translation ▪ Not robust against noise ▪ Modification of quantization table



http://www.abbascheddad.net/Survey.pdf

Image Steganography (cont.)

Example: Comparison of Different Tools for Image Steganography

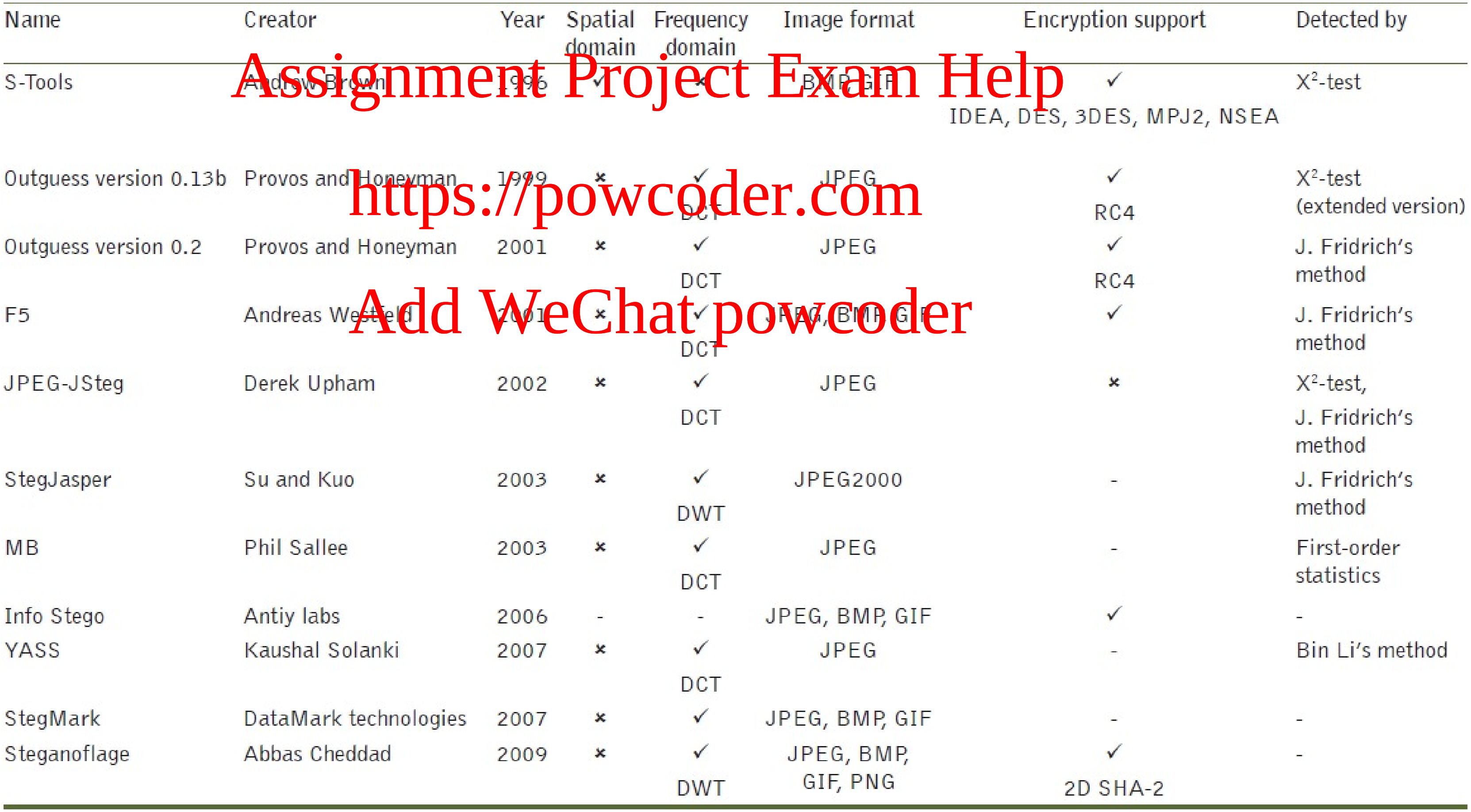


			space	frequency	encryption	random.		
Name	Creator	Year	(1)	(2)	(3)	(4)	(5)	Detected by
JSteg	Derek Upham	-	x	✓ DCT	x	x	JPEG	- X^2-test - Stegdetect -Fridrich's Algorithm
JSteg-Shell	John Korejwa	-	x	✓ DCT	✓ RC4	-	JPEG	- X^2-test
OutGuess version 0.13b	Provos and Honeyman	-	x	✓ DCT	✓ RC4	✓	JPEG	- X^2-test (extended version) - Stegdetect
White Noise Storm	Ray (Arsen) Arachelian	1994	✓	x	✓	✓	PCX	- X^2-test
EZStego	Romana Machado	1996	✓	x	✓	x	BMP, GIF	-RS-steganalysis
S-Tools	Andrew Brown	1996	✓	x	✓ IDEA, DES, 3DES,MPJ2, NSEA	x	BMP, GIF	- X^2-test
JPhide	Allan Latham	1999	x	✓ DCT	✓ Blowfish	x	JPEG	- X^2-test - Stegdetect
OutGuess version 0.2	Provos and Honeyman	2001	x	✓ DCT	✓ RC4	✓	JPEG	-Fridrich's Algorithm
F5	Andreas Westfeld	2001	x	✓	✓	✓	JPEG	-Fridrich's Algorithm

http://www.abbascheddad.net/Survey.pdf

Image Steganography (cont.)

Example: Comparison of Different Tools for Image Steganography



Name	Creator	Year	Spatial domain	Frequency domain	Image format	Encryption support	Detected by
S-Tools	Andrew Brown	1996	✓	×	BMP, GIF	✓ IDEA, DES, 3DES, MPJ2, NSEA	X²-test
Outguess version 0.13b	Provos and Honeyman	1999	×	✓ DCT	JPEG	✓ RC4	X²-test (extended version)
Outguess version 0.2	Provos and Honeyman	2001	×	✓ DCT	JPEG	✓ RC4	J. Fridrich's method
F5	Andreas Westfeld	2001	×	✓ DCT	JPEG, BMP, GIF	✓	J. Fridrich's method
JPEG-JSteg	Derek Upham	2002	×	✓ DCT	JPEG	×	X²-test, J. Fridrich's method
StegJasper	Su and Kuo	2003	×	✓ DWT	JPEG2000	-	J. Fridrich's method
MB	Phil Sallee	2003	×	✓ DCT	JPEG	-	First-order statistics
Info Stego	Antiy labs	2006	-	-	JPEG, BMP, GIF	✓	-
YASS	Kaushal Solanki	2007	×	✓ DCT	JPEG	-	Bin Li's method
StegMark	DataMark technologies	2007	×	✓	JPEG, BMP, GIF	-	-
Steganoflage	Abbas Cheddad	2009	×	✓ DWT	JPEG, BMP, GIF, PNG	✓ 2D SHA-2	-

http://www.tr.ietejournals.org/viewimage.asp?img=IETETechRev_2013_30_4_344_116724_t13.jpg