Image Steganography using LSB

ELL205 Course Project

Shashank Goyal - 2018EE10499

Rohit Agrawal - 2018EE10494

Introduction

- Nowadays most of the data is stored digitally. Spreading of information has extensively increased since the development of the internet and millions of megabytes of data travel over the networks every day.
- Some of this data is personal and very sensitive like passwords, atm pins etc. which attracts a lot of attention.
- Therefore preserving, confidentiality and data integrity against unauthorized access is very important.
- Due to these security issues there are various ways that have been developed for securing information.

Steganography

- The term Steganography is a combination of Greek words **stegnos** which means "covered or protected" and **grafia** means "writing"; so steganography itself means **"concealed writing"**.
- Steganography is the science of hiding a message in a manner such that the existence of message is known only to the recipient of the message.
- The aim of steganography is to go incognito i.e. if no attention is drawn to the hidden message then steganography has achieved its goal.

Why Steganography not Cryptography

- Cryptography changes the message into an unreadable format. This process is known as encryption; only intended user can convert it into a normal message.
- But this might generate the curiosity of the intruder as the encrypted message is available. A message in plain sight which is easily available will definitely attract the curiosity of a potential intruder as it will set him up with the challenge of cracking the code and generating sense out of the garbage message.
- It might also plant the idea that the message must be something confidential as someone has taken the effort to write it in a code that can only be cracked by someone holding the key, as opposed to a hidden message which might go undetected and hence reducing the risk of attempts being made to decipher it.

LSB Algorithm

- This is a simple logic in which the message is broken into 1 bit pieces and the least significant bit of each pixel of the cover object is used to carry the secret message pieces.
- The method can't be easily countered as the change in cover object is not visible to the naked eye as we are just changing the value by ±1.

LSB Algorithm

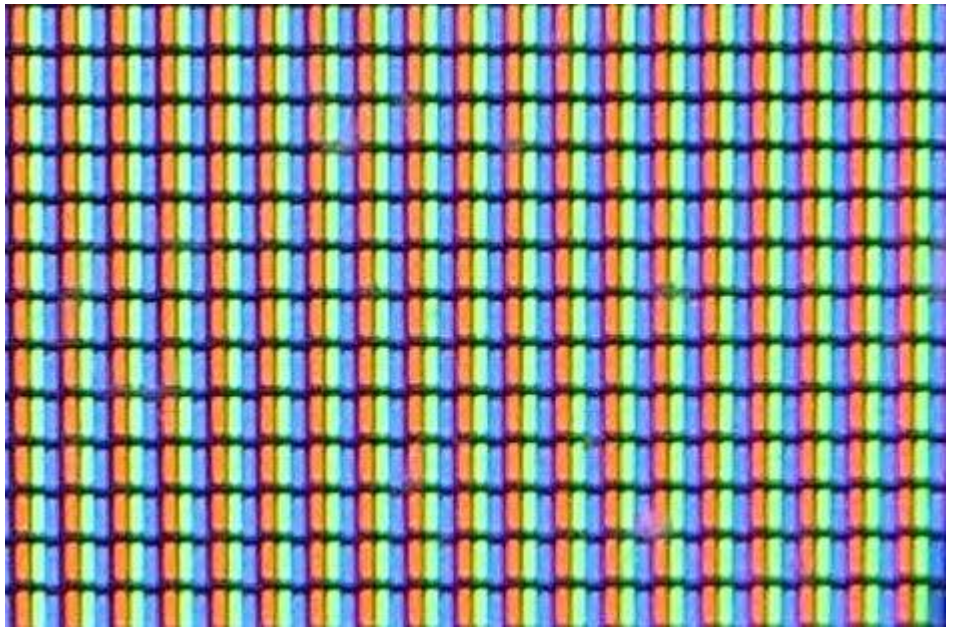
- We first break the message in characters. Then we take the ASCII values of those characters and convert them to 8 bit binary values.

"ok" -> 111 107

111 107 -> 01101111 01101011

LSB Algorithm

- We iterate through the pixels of the image in sets of three pixels



LSB Algorithm

- We iterate through the pixels of the image in sets of three pixels

- We have 9 values in a set – 3 R, 3 G and 3 B.

	Pixel 1	Pixel 2	Pixel 3
R	97	105	102
G	167	172	158
B	230	231	228

LSB Algorithm

- We change each value to even or odd according to the next bit. Even for 0 and odd for 1
  - "ok" -> 111 107

111 107 -> 01101111 01101011

	Pixel 1	Pixel 2	Pixel 3
R	97	105	102
G	167	172	158
B	230	231	228

LSB Algorithm

- We change each value to even or odd according to the next bit.
  Even for 0 and odd for 1
  - "ok" -> 111 107

  111 107 -> 01101111 01101011

	Pixel 1	Pixel 2	Pixel 3
R	96	105	102
G	167	172	158
B	230	231	228

LSB Algorithm

- We change each value to even or odd according to the next bit. Even for 0 and odd for 1
  - "ok" -> 111 107

111 107 -> 01101111 01101011

	Pixel 1	Pixel 2	Pixel 3
R	96	105	102
G	167	172	158
B	230	231	228

LSB Algorithm

- We change each value to even or odd according to the next bit. Even for 0 and odd for 1
  - "ok" -> 111 107

111 107 -> 01101111 01101011

	Pixel 1	Pixel 2	Pixel 3
R	96	105	102
G	167	172	158
B	230	231	228

LSB Algorithm

- We change each value to even or odd according to the next bit.
  Even for 0 and odd for 1
  - "ok" -> 111 107

  111 107 -> 01101111 01101011

	Pixel 1	Pixel 2	Pixel 3
R	96	105	103
G	166	173	159
B	231	231	228

LSB Algorithm

- For the ninth value, we make it even if the message if finished else odd.
  - "ok" -> 111 107

111 107 -> 01101111 01101011

	Pixel 1	Pixel 2	Pixel 3
R	96	105	103
G	166	173	159
B	231	231	228

LSB Algorithm

- ‘o’ is encoded in the image
- “ok” -> 111 107

111 107 -> 01101111 01101011

	Pixel 1	Pixel 2	Pixel 3
R	96	105	103
G	166	173	159
B	231	231	228

LSB Algorithm

- Doing the same for 'k'
- "ok" -> 111 107

111 107 -> 01101111 01101011

	Pixel 1	Pixel 2	Pixel 3
R	96	105	103
G	166	173	159
B	231	231	228

	Pixel 1	Pixel 2	Pixel 3
R	156	155	153
G	126	123	126
B	31	33	35

What more we can do?

- We thought of ways to make our message more secure while still being hidden in plain sight.
- The problem with plain steganography is that if an attacker has our decoding code or knows the bit manipulation pattern, the message can decoded within seconds.

Our Solution

- We thought of incorporating a password or a key in our algorithm.
- We came up with a way to store information such that even if our code or full knowledge of our algorithm is available, the attacker cannot decode the message without the password.
- This also neither affects the visibility of our message nor the maximum message length capacity of our image.

How it works?

- We take in an string input during encoding and from it we create a vector containing values 1 to 8 in jumbled order.

hello -> [8 2 3 1 4 5 6 7]

- Now rather than storing message in a linear fashion, we follow the jumbled order produced.
- Initially the bit of each character of message stored in a pixel was as follows -

	Pixel 1	Pixel 2	Pixel 3
R	1	2	3
G	4	5	6
B	7	8	E

How it works?

- We take in an string input during encoding and from it we create a vector containing values 1 to 8 in jumbled order.

hello -> [8 2 3 1 4 5 6 7]

- Now rather than storing message in a linear fashion, we follow the jumbled order produced.
- Now the bit of character stored are in this order

	Pixel 1	Pixel 2	Pixel 3
R	8	2	3
G	1	4	5
B	6	7	E

How it works?

- Decoder takes a key as input to produce the same pattern for decode the image.
- Any wrong key given as input will lead to random order and thus will yield a garbage output.

Challenges faced

- The total number or patterns possible are 8! = 40320 only.
- jpeg cannot be used as the output image format as the message is lost because of its lossy nature.
- The maximum message length that can be stored is fixed for an image.

Thank You!