
COMP 8505

Assignment 2 - Steganography

Geoff Dabu - Oct 5, 2015

COMP 8505	1
Assignment 2 - Steganography	1
Introduction	3
Design	4
Tests	6

Introduction

The purpose of this assignment is to design and develop a program which uses the LSB steganography method to encode and decode data from an image. Usually, each pixel of an image has at least 24 bits. 8 bits per Red, Green and Blue value. The LSB method works by replacing the least significant bit with data.

Design

Pseudo Code

stego_write.py

load image

load secret file and covert into string of bits

add file name and file size to the beginning of the string of bits for the secret file

the file name will be followed by a null char

the file size will be followed by a null char

loop through every pixel in image

 get r g b values of pixel

 set last bit of r to a bit from the secret file

 set last bit of g to a bit from the secret file

 set last bit of b to a bit from the secret file

 exit loop when there are no bits left to store

save image as cover image

sego_read.py

load cover image

loop through every pixel in image

- get rgb values of pixel
- get last bit of r
- get last bit of g
- get last bit of b

check for file name and file size **note: file size is the number of bits in the secret data

- after 8 bits are retrieved check to see if it is a null char

- after first null char
 - the previous bits are your file name

- after second null char
 - the bits in-between the first and second null char are your data
 - stop checking for file name and file size
 - start retrieving the data

start retrieving data

- loop through the rest of the pixels
 - store bits into data bitString
 - exit loop when you get all the bits required

save data bitString to a file with the name you retrieved earlier.

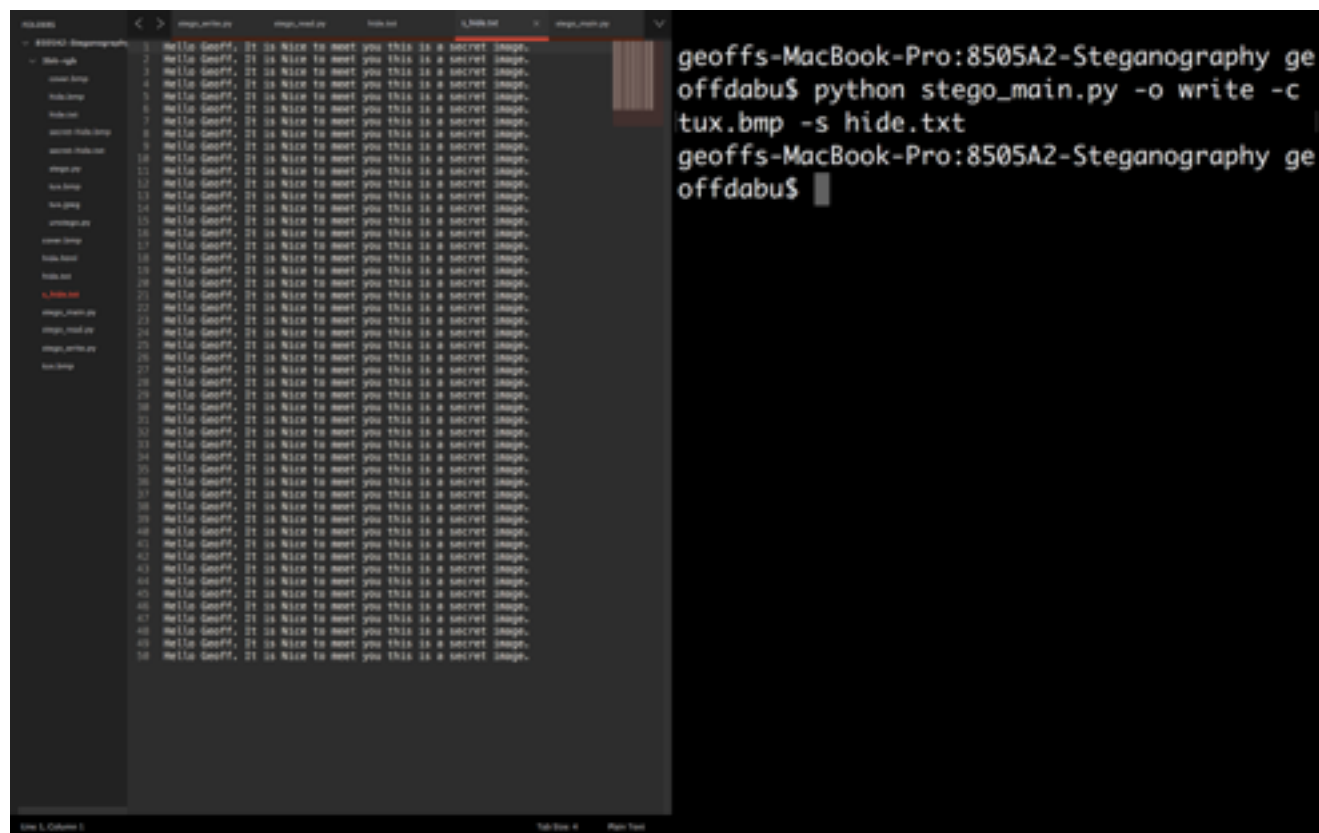
Tests

Screenshots of output files will be used during these tests to prove or disprove functionality of this program.

BMP Cover Test - Testing BMPs as cover images.

Test 1.

Cover Image File Type: BMP

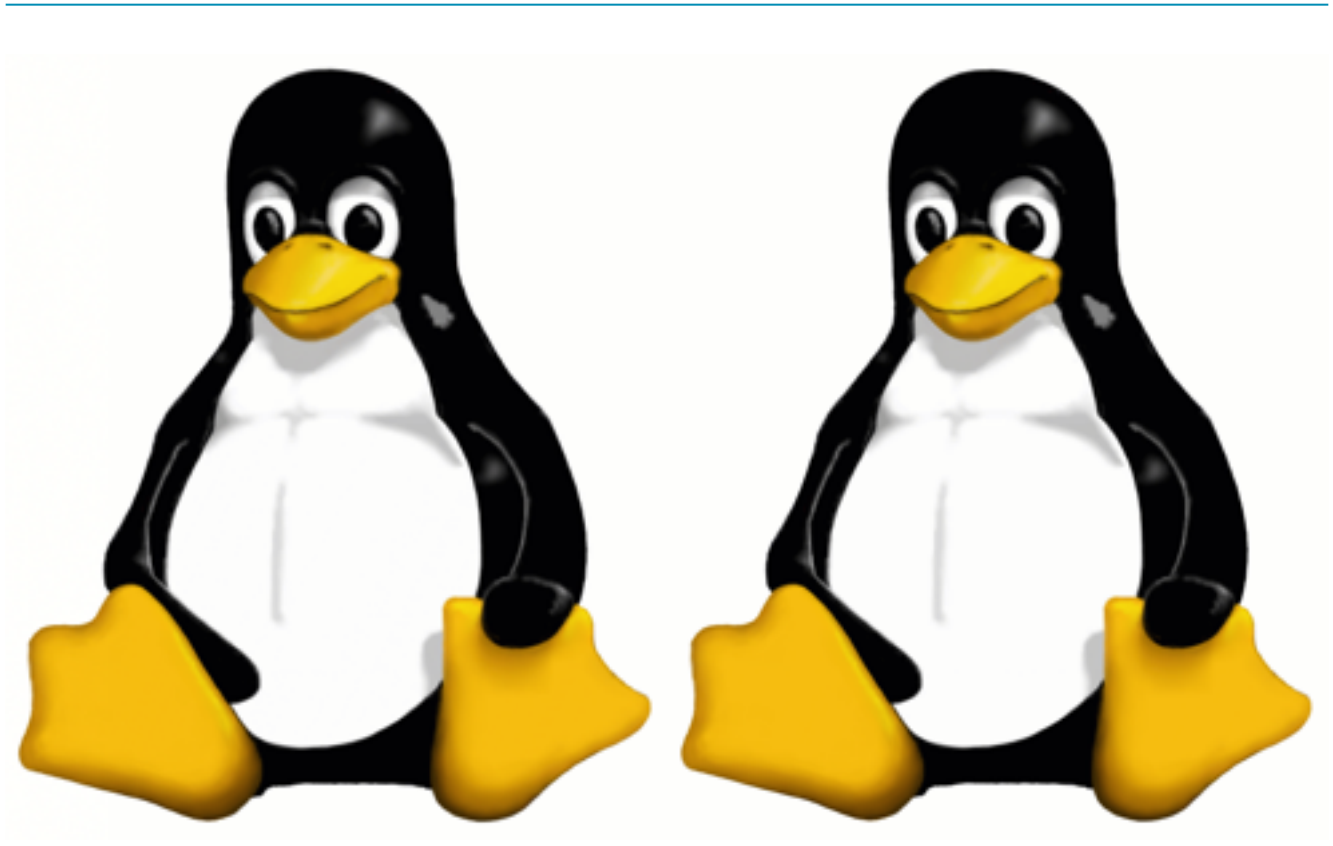


Secret Image File Type: TXT

The image above shows the text file which was hidden in the cover image, and the command that was used to execute the `stego_write` command.

```
geoffss-MacBook-Pro:8505A2-Steganography geoffdabu$ python stego_main.py -o write -c tux.bmp -s hide.txt
geoffss-MacBook-Pro:8505A2-Steganography geoffdabu$ python stego_main.py -o read -c cover.bmp
geoffss-MacBook-Pro:8505A2-Steganography geoffdabu$
```

The image above shows the output text file that was produced after running the stego_read on the cover image.



The image on the left is the original BMP image, whereas the image on the right is the cover image which has been modified using LSB to encapsulate the txt file.

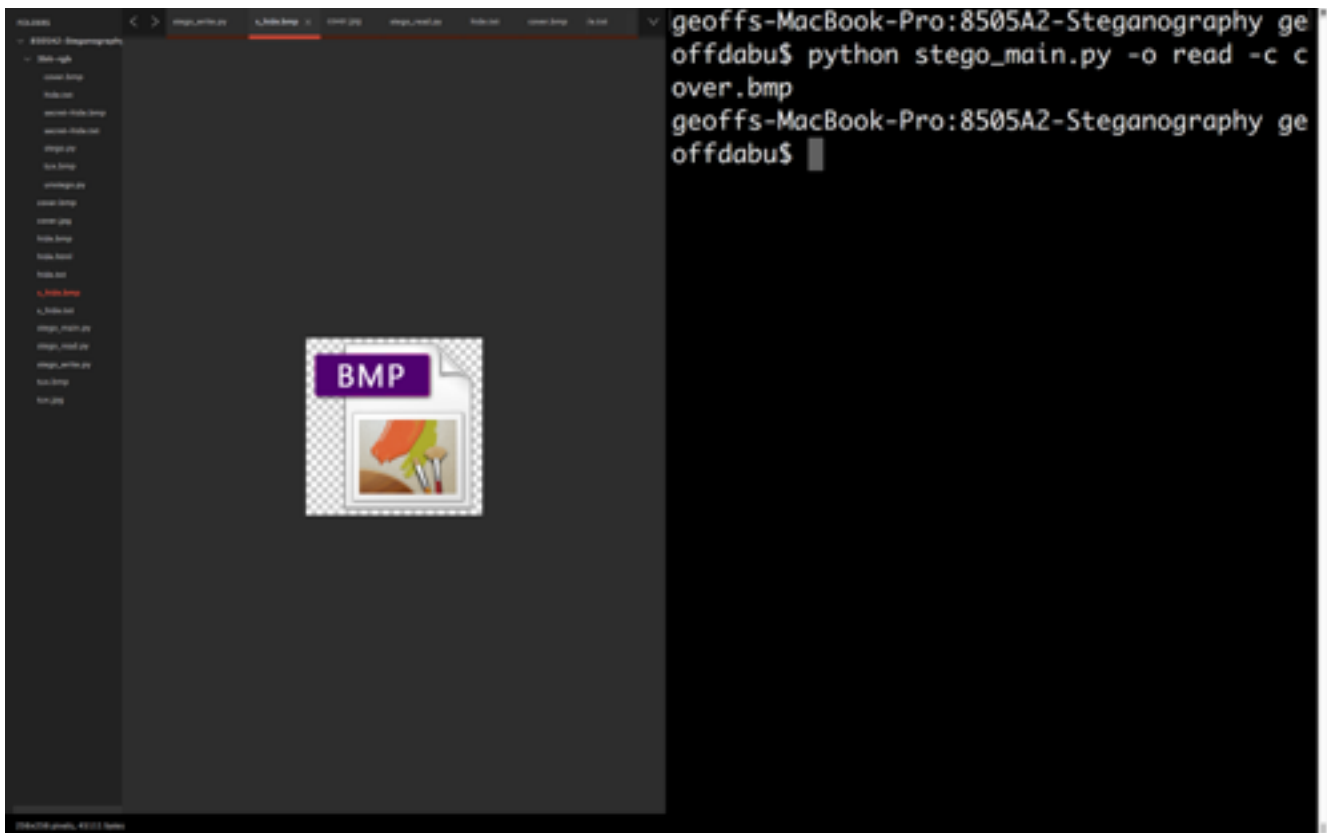
Test 2.

Cover Image File Type: BMP

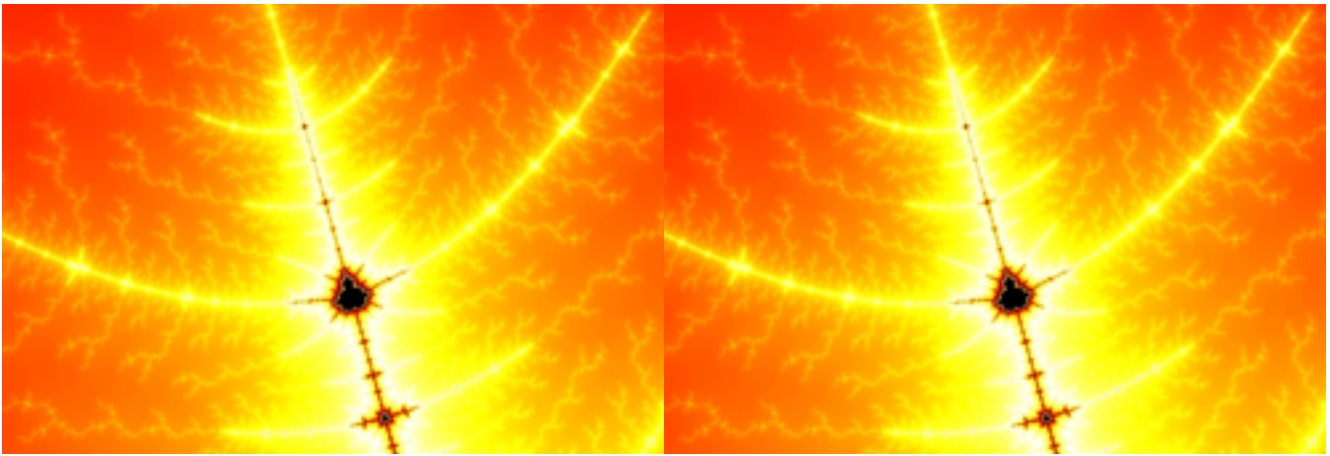
Secret Image File Type: BMP



The image above shows the bmp file which was hidden in the cover image, and the command that was used to execute the stego_write program.



The image above shows the output text file that was produced after running the `stego_read` on the cover image.



The image on the left is the original BMP image, whereas the image on the right is the cover image which has been modified using LSB to encapsulate the bmp file.

Size Test - Since it takes 8 bytes of the cover image to hide 8 bits of the data. It is important that the cover image is large enough to accommodate the size of the secret file.

The equation for this would be:

$$\text{cover image width} * \text{cover image height} * 3 \geq \text{number of bits being encoded}$$

Test 1.

Cover Image Size : 800px * 950px = 760,000px

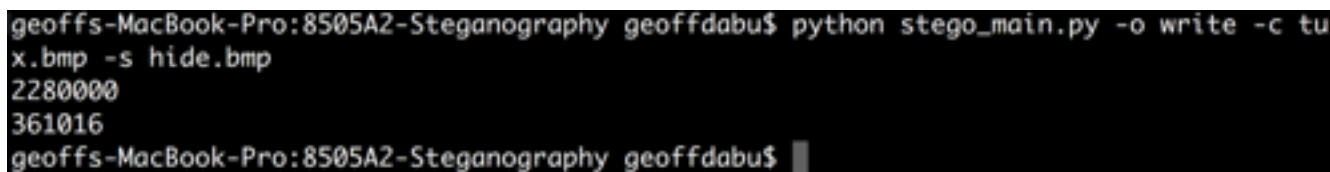
Number of encoded bits: 361016 bits

760,000 * 3 = 2,280,000 is greater than 361016

Expected Outcome: File should be fully encoded.

Actual Outcome: Entire file was encoded.

Screenshot:



```
geofffs-MacBook-Pro:8505A2-Steganography geoffdabu$ python stego_main.py -o write -c tux.bmp -s hide.bmp
2280000
361016
geofffs-MacBook-Pro:8505A2-Steganography geoffdabu$
```

Test 2.

Cover Image Size : 256px * 256px = 65,536px

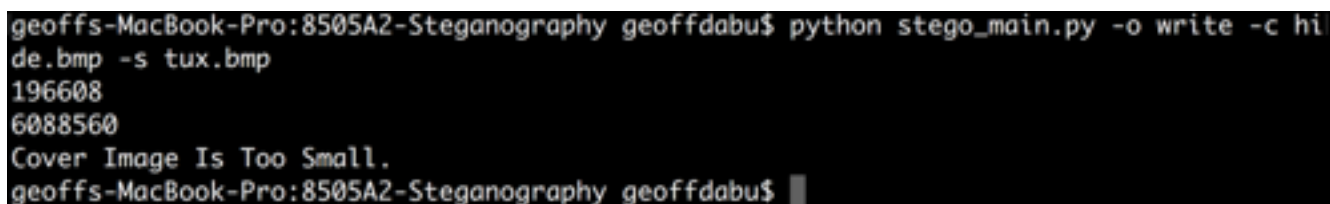
Number of encoded bits: 6,088,560 bits

65,536 * 3 = 196,608 is less than 60,088,560

Expected Outcome: Program should not encode, and should exit.

Actual Outcome: Program exited, due to cover image being too small.

Screenshot:



```
geofffs-MacBook-Pro:8505A2-Steganography geoffdabu$ python stego_main.py -o write -c hide.bmp -s tux.bmp
196608
6088560
Cover Image Is Too Small.
geofffs-MacBook-Pro:8505A2-Steganography geoffdabu$
```