**Image Steganography**

Saket Singh Rachit Jain

140905450 140905464

Roll no.: 39 Roll no.: 40

**ABSTRACT**

Steganography is the practice of concealing a file, message, image, or video within another file, message, image, or video. Whereas cryptography is the practice of protecting the contents of a message alone, steganography is concerned with concealing the fact that a secret message is being sent, as well as concealing the contents of the message.

This project will implement image steganography for bitmap images that is efficient by processing parts of an image in parallel instead of processing them sequentially, which would result in considerably faster processing rates.

**MOTIVATION**

Information is not safe. Hiding a secret message can secure information. It is even better if no one knows that a hidden message is being sent. This is where steganography comes into place. It is very difficult to detect if a secret message is being sent because there is a very minute difference between the original image and the message encoded image. The difference between the two images is so small that it is not visible to the naked eye. If a program is run, only then can someone realize the difference.

**OBJECTIVES**

A secret message will be hidden in an image by an encoding function. The new image will contain the hidden message stored as a part of the blue tint values of the original image. The new image can be decoded to find out the secret message which was encoded by using a decoding function.

**METHODOLOGY**

In a bitmap image, the image information is stored as consecutive bits. We will be concealing the message by overwriting the last 2 bits of each byte/2 bytes. By changing the least significant bits, we are ensuring that the image is not drastically different from the original.

To recover the hidden message, we can read the image and extract the last two bits of each byte and concatenate them to form the concealed original message/file.

One way of figuring out how the message stored is not visible to the naked eye, is to take the bitwise XOR of the original image and the encrypted image.

**Implementation on MPI**

The image will be read once and will be shared by all the sub-processes. Each process will carry out the desired operations on a single row of the image matrix. After the processing is done, the root process will recollect the buffer data from all the child processes to reconstruct the final image.

**Implementation on OpenCL/CUDA**

Multiple kernels will be defined to work on specific tasks. The input matrix will be divided into sub-matrices which will then be passed to the processors, which will execute the desired kernel. All the work items will write to a chunk of memory which will be read back by the host device. Kernels will be written to perform encryption, decryption and to find the bitwise difference between two images.

**RESULTS**

We were able to achieve faster results when compared to a sequential code for the same. MPI creates processes in which each process processes a substring of the original data, divided equally according to the number of instances. It resulted in storing the message in the image with a better speed. CUDA uses the GPU for storing the message in the image. A GPU based approach to this problem resulted in even faster speeds when compared to MPI.

**ALGORITHM**

**Encoding -**

*Execute in parallel for all characters   
c <- character to be encoded  
while c is not consumed*

*NewBlueValue <- (OldBlueValue >> 2) << 2 | (c&0x3)  
c <- (c >> 2)*

*return NewBlueValue*

**Decoding -**

*Execute in parallel for 4 blue tints at once   
c <- character to be decoded  
j <- 0  
pos <- starting position of character data  
do*

*hidden <- NewBlueValue & 0x3  
c <- (c | hidden << (j \* 2))*

*while j < 4*

*return c*

**CONCLUSION**

Hiding a message in an image is a unique way of encrypting data thus hiding the data in plain sight. Using a parallel programming based approach resulted in much faster processing rates. With the help of parallel programming large amount of data can be stored in an image very quickly as all the computationally expensive part is shifted to a process (MPI) and kernels (CUDA).

**REFERENCES**

1). <https://en.wikipedia.org/wiki/Steganography>

2). <http://stackoverflow.com/questions/15651567/steganography-in-c>

3). Parallel Programming in C with MPI and OpenMPI