**F28HS 2019-2020**

**Hardware-Software Interface**

**C Coursework**

**Hasan Kapadia**

**H00289018**

# **Introduction:**

This coursework was based upon us creating an algorithm to encode and decode images using the **PPM** image extension to hide secret messages.

Each pixel of an image consists of bits where each bit gives the color needed for an image.

Images consist of three major values which make up the different shades/gradients of the image

RGB ranked from 0 to 255 where each value gives the image different shades of color .

For e.g. -

R G B

Red- (255,0,0) bit representation would be- 111111110000000000000000

Green (0,255,0) bit representation would be- 000000001111111100000000

Blue (0,0,255) bit representation would be- 000000000000000011111111

# **Programme Design:**

My Programme design is based of the LSB (Least Significant Bit) Technique which comprises of replacing the least significant bits (Rightmost bit) in an image file with the secret message.

# **Code Logic:**

My code (although incomplete) was written using the logic that an image is taken, and symmetric key algorithm is used to make it so that users must use a single key to encrypt or decrypt the image.

In this code the image is taken and converted into binary bits where the LSB technique is used to change the right most bits (least significant bits) of each pixel in an image to encode a secret message into them

This is used in par with the symmetric key algorithm to add a sense of security where to decode the message a pass key is required from the user and the key should match the key inputted into the system

To make the encode work the srand() function of c is used where srand() is a pseudo-random number generator where it is used with the time\_t function to generate a different random string of numbers each and every time

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

//structs

struct PPM

{

    char LetterCode[2];

    int Width;

    int Height;

    int max;

    int Val[];

    struct values \* value;

};

struct values

{

    int R;

    int G;

    int B;

};

//functions

//return a C structure PPM image from file “f”

struct PPM \* getPPM(FILE \* f);

//display the PPM image “im” as text

void showPPM(struct PPM \* im);

//encode

struct PPM \* encode (struct PPM \* im, char \* message, unsigned int mSize, unsigned int secret);

//decode

char \* decode(struct PPM \* im, unsigned int secret);

//main function

int main(int args, char \*\* argv) {

    FILE \*f;

    FILE \*f2;

    f = fopen("star.ppm","r");

    struct PPM \* ppm = getPPM(f);

    struct PPM \* encodedPPM = encode(ppm, "HSI", strlen("HSI"), 2560867);

    char \* message = decode(encodedPPM, 2560867);

    printf("\n%s\n", message);

    fclose(f);

    return 0;

}

struct PPM \* getPPM(FILE \* f)

{

}

void showPPM(struct PPM \* im)

{

    printf("%s\n", im->LetterCode);

    printf("%d %d\n%d\n", im->Width, im->Height, im->max);

    int size = im->Width \* im->Height \* 3;

    int i = 0;

    while(i < size)

    {

        printf("%d %d %d\n", im->Val

        [i], im->Val

        [i+1], im->Val

        [i+2]);

        i+=3;

    }

}

//encode refrenced from github https://github.com/nyalsadiq/Steganography/blob/master/F28HS.c

struct PPM \* encode (struct PPM \* im, char \* message, unsigned int mSize, unsigned int secret)

{

int i,j,start;

    int counter = 0;

    time\_t t;

    srand((unsigned) time(&t));

    start = rand() % (im->Height\*im->Width/ 2);

    for(i = start; i < (im->Height\*im->Width); i++){

            if (counter < strlen(message)){

                im->value[i].R = message[counter];

                counter = counter + 1;

            }

            if(counter < strlen(message)) {

                im->value[i].G = message[counter];

                counter = counter + 1;

            }

            if(counter < strlen(message)) {

                im->value[i].B = message[counter];

                counter = counter + 1;

            }

            else{

                return im;

            }

    }

    return im;

}

char \* decode(struct PPM \* im, unsigned int secret)

{

}