In our project we used quite simple approach, we are encoding one byte of message information on the ‘younger’ bits of one pixel bytes, last 3, last 3 and last 2 bits of RGB, respectively. Moreover it’s needed to encode information of how many bytes of information we are encoding. The length of the message is converted into string and written to pixels just like the actual message. Moreover it’s needed to encode information about how long is that length of message to read, that’s the first information encoded in pixels. The problems related to that kind of implementation are as follows:

1) if we encode pixels one by one then it can be seen that something changes in the picture, especially when it comes to white color; one possible solution for that is writing information to pixels with some offset, yet with that solution the next problem occurs

2) when message is longer than size of picture divided by 3 then it’s not possible to encode information on younger bits like written above; the possible solution for that is encoding information into older bits, yet that is risky due to possibility of changing picture in some more visible way; other solution is simply giving a strict number of characters which may be encoded.

To compile a program, the following command should be used:
gcc stego.c -o stego

To run application, following command are proper for encoding and decoding, respectively:
./stego h <secret_msg_file> <image_file> <new_image_file>
./stego u <image_file> <recovered_secret_msg>

ex.
./stego h juliuscaeser.txt taz.ppm code.ppm

Authors:
Joanna Rajewska 42287
Łukasz Wojtczak 42293