WOJSKOWA AKADEMIA TECHNICZNA
im. Jarosława Dąbrowskiego
WYDZIAŁ CYBERNETYKI

Steganografia
Lab. 4

Student
x

Prowadzący laboratoria:
Y

Spis treści

Treść zadania

1. Napisać skrypt w programie Matlab wczytujący obraz o 24-bitowej głębi koloru.
2. Skrypt ma wyodrębniać kolejne warstwy najmniej znaczących bitów pikseli obrazu i wizualizować je. Należy zwizualizować minimum 3 najmniej znaczące bity dla każdego koloru.
3. W obrazie o 24-bitowej głębii koloru, z wykorzystaniem algorytmu LSB dokonać ukrycia obrazu o 256 odcieniach szarości.
4. Zaprezentować porównanie warstw koloru pomiędzy obrazem oryginalnym a zmodyfikowanym

Kod realizujący zadanie

Zadanie zrealizowano w postaci jednego skryptu lab4_Miazga.m

Poniżej zaprezentowano zawartość skryptu wraz z komentarzami:

lab4_Miazga.m

```m
%wczytujanie obrazu o 24-bitowej głębi koloru

%w poniższym obrazie będzie ukrywany drugi obraz (o 256 odcieniach szarości)
image_1 = imread('eminem-rihanna.jpg');%eminem-rihanna.jpg
[height_1, width_1, layers_1] = size(image_1);

%wczytanie obrazu o 256 odcieniach szarości
%poniższy obraz będzie ukrywany w pierwszym obrazie
image_2 = imread('cat_eye.jpg');
[height_2, width_2, layers_2] = size(image_2);

%wyodrębnienie kolejnych warstw najmniej znaczących bitów pikseli obrazu
%warstwy zostaną zapisane (zwizualizowane) jako obrazy
%operacje zostaną wykonane dla 3 najmniej znaczących bitów dla każdego koloru.

%allBlack to tabela rozmiaru obrazu wypełniona samymi zerami
allBlack = zeros(height_1, width_1, 'uint8');

%temp_matrix to 3 warstwy odpowiadające wielkościom warstw obrazu,wypełnione zerami
temp_1_matrix = zeros(height_1, width_1, layers_1);
temp_2_matrix = zeros(height_1, width_1, layers_1);
temp_3_matrix = zeros(height_1, width_1, layers_1);

for i = 1 : 1 : height_1
    for j = 1 : 1 : width_1
        for k = 1 : 1 : layers_1
            pix_number = image_1(i,j,k);
            bin = de2bi(pix_number,8);

            LSB_1 = bin(1); %pierwszy najmniej znaczący bit

            LSB_2 = bin(2); %drugi najmniej znaczący bit
            LSB_3 = bin(3); %trzeci najmniej znaczący bit

            if LSB_1 == 1
               temp_1_matrix(i,j,k) = 255;
            end
            if LSB_2 == 1
               temp_2_matrix(i,j,k) = 255;
            end
            if LSB_3 == 1
               temp_3_matrix(i,j,k) = 255;
            end
        end
    end
end

% first LSB
layer_r_1 = cat(3,temp_1_matrix(:,:,1),allBlack,allBlack);
layer_g_1 = cat(3,allBlack,temp_1_matrix(:,:,2),allBlack);
layer_b_1 = cat(3,allBlack,allBlack,temp_1_matrix(:,:,3));

% secound LSB
layer_r_2 = cat(3,temp_2_matrix(:,:,1),allBlack,allBlack);
layer_g_2 = cat(3,allBlack,temp_2_matrix(:,:,2),allBlack);
layer_b_2 = cat(3,allBlack,allBlack,temp_2_matrix(:,:,3));

% third LSB
layer_r_3 = cat(3,temp_3_matrix(:,:,1),allBlack,allBlack);
layer_g_3 = cat(3,allBlack,temp_3_matrix(:,:,2),allBlack);
layer_b_3 = cat(3,allBlack,allBlack,temp_3_matrix(:,:,3));
```

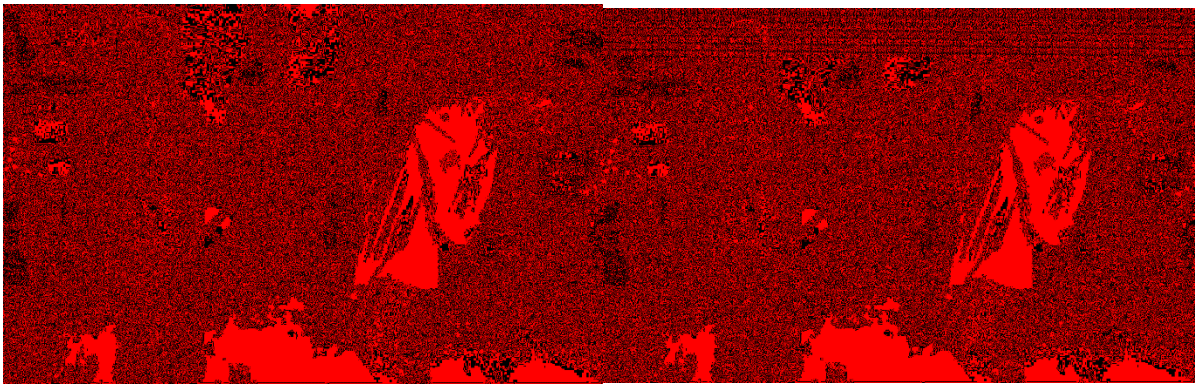
```
61
62      %zapisanie wyodrębnionych warstw
63      imwrite(layer_r_1,"layer_r_1.png");
64      imwrite(layer_g_1,"layer_g_1.png");
65      imwrite(layer_b_1,"layer_b_1.png");
66
67      imwrite(layer_r_2,"layer_r_2.png");
68      imwrite(layer_g_2,"layer_g_2.png");
69      imwrite(layer_b_2,"layer_b_2.png");
70
71      imwrite(layer_r_3,"layer_r_3.png");
72      imwrite(layer_g_3,"layer_g_3.png");
73      imwrite(layer_b_3,"layer_b_3.png");
74
75      %wiem, że obraz drugi jest mniejszy niż pierwszy, natomiast gdybym pisał
76      %kod na potrzeby powtarzania operacji na innych obrazach trzeba by sprawdzać
77      %warunek, czy obraz, który ukrywamy zmieści się w obrazie, w którym chcemy
78      %ukrywać
79
80      %ukrycie obrazu drugiego w obrazie pierwszym za pomocą algorytmu LSB
81
82
83      % warstw oba obrazy mają tyle samo i dla tych obrazów są one tej samej wielkości,
84      % problem mozna uprościć do zapisania drugiego obrazu w pierwszym dla jednej warstwy, pozostałe zrobi się analogicznie
85      row = 1;
86      col = 1;
87      counter = 0;
88      for warstwa = 1 : 1 : 3
89          for i = 1 : 1 : height 2
```

```
90              for j = 1 : 1 : width_2
91                      number = image_2(i,j,1);
92                      bin_number = de2bi(number,8);
93                          for l = 1 : 1 : 8
94                              LSB = mod(double(image_1(row,col,warstwa)), 2);
95                              temp = double(xor(LSB, bin_number(l)));
96                              image_1(row,col,warstwa) = image_1(row,col,warstwa) + temp;
97                              col = col + 1;
98                              if(col > width_1)
99                                  col = 1;
100                                 row = row + 1;
101                             end
102                             counter = counter + 1;
103                         end
104
105             end
106         end
107     end
108
109     %ekstrakcja bitów ukrytych - na potrzeby sprawdzenia
110     %row = 1;
111     %col = 1;
112     %for i = 1 : 1 : 1560
113     %    extr_bits(i,1) = mod(double(image_1(row,col,1)), 2);
114     %    col = col + 1;
115     %    if(col > width_1)
116     %        col = 1;
117     %        row = row + 1;
118     %%    end
```

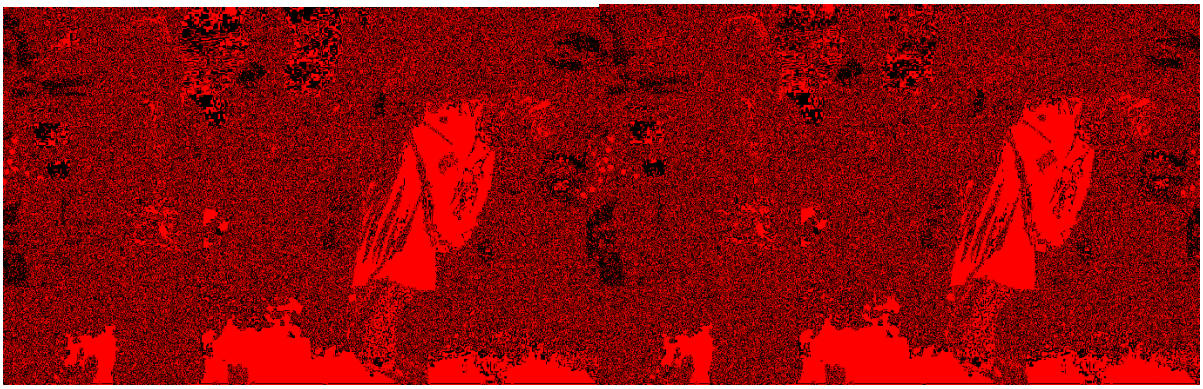
```
%end
%disp('extracted bits')
%disp(extr_bits)

temp_1_matrix_v2 = zeros(height_1, width_1, layers_1);
temp_2_matrix_v2 = zeros(height_1, width_1, layers_1);
temp_3_matrix_v2 = zeros(height_1, width_1, layers_1);

for i = 1 : 1 : height_1
    for j = 1 : 1 : width_1
        for k = 1 : 1 : layers_1
            pix_number = image_1(i,j,k);
            bin = de2bi(pix_number,8);

            LSB_1 = bin(1); %pierwszy najmniej znaczący bit
            LSB_2 = bin(2); %drugi najmniej znaczący bit
            LSB_3 = bin(3); %trzeci najmniej znaczący bit

            if LSB_1 == 1
                temp_1_matrix_v2(i,j,k) = 255;
            end
            if LSB_2 == 1
                temp_2_matrix_v2(i,j,k) = 255;
            end
            if LSB_3 == 1
                temp_3_matrix_v2(i,j,k) = 255;
            end
        end
    end
end

% first LSB
layer_r_1_v2 = cat(3,temp_1_matrix_v2(:,:,1),allBlack,allBlack);
layer_g_1_v2 = cat(3,allBlack,temp_1_matrix_v2(:,:,2),allBlack);
layer_b_1_v2 = cat(3,allBlack,allBlack,temp_1_matrix_v2(:,:,3));

% secound LSB
layer_r_2_v2 = cat(3,temp_2_matrix_v2(:,:,1),allBlack,allBlack);
layer_g_2_v2 = cat(3,allBlack,temp_2_matrix_v2(:,:,2),allBlack);
layer_b_2_v2 = cat(3,allBlack,allBlack,temp_2_matrix_v2(:,:,3));

% third LSB
layer_r_3_v2 = cat(3,temp_3_matrix_v2(:,:,1),allBlack,allBlack);
layer_g_3_v2 = cat(3,allBlack,temp_3_matrix_v2(:,:,2),allBlack);
layer_b_3_v2 = cat(3,allBlack,allBlack,temp_3_matrix_v2(:,:,3));

%zapisanie wyodrębnionych warstw
imwrite(layer_r_1_v2,"layer_r_1_v2.png");
imwrite(layer_g_1_v2,"layer_g_1_v2.png");
imwrite(layer_b_1_v2,"layer_b_1_v2.png");

imwrite(layer_r_2_v2,"layer_r_2_v2.png");
imwrite(layer_g_2_v2,"layer_g_2_v2.png");
imwrite(layer_b_2_v2,"layer_b_2_v2.png");

imwrite(layer_r_3_v2,"layer_r_3v.png");
imwrite(layer_g_3_v2,"layer_g_3_v2.png");
imwrite(layer_b_3_v2,"layer_b_3_v2.png");
```

Porównanie warstw koloru pomiędzy obrazem oryginalnym a zmodyfikowanym

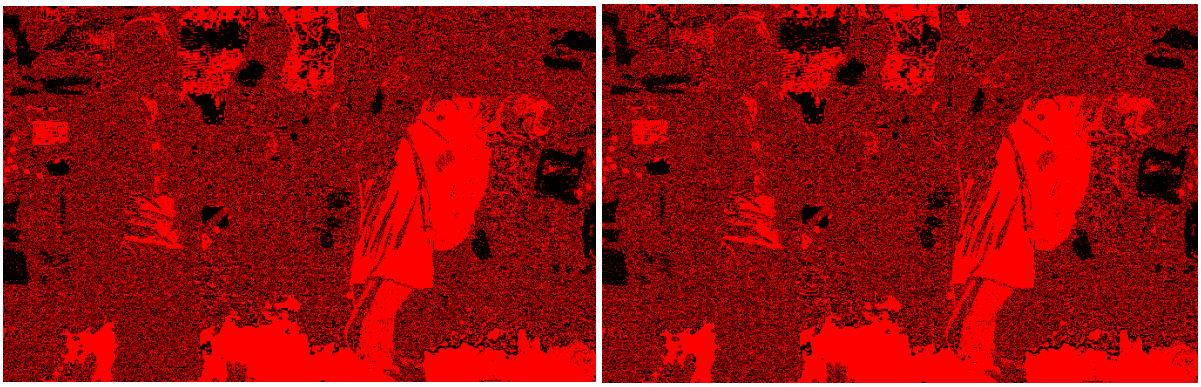
Warstwa czerwona, pierwszy najmniej znaczący bit



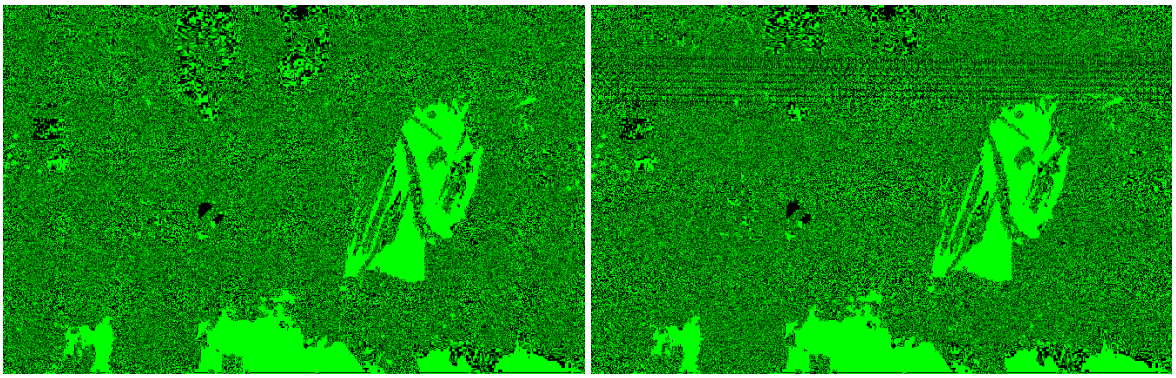
Warstwa czerwona, drugi najmniej znaczący bit



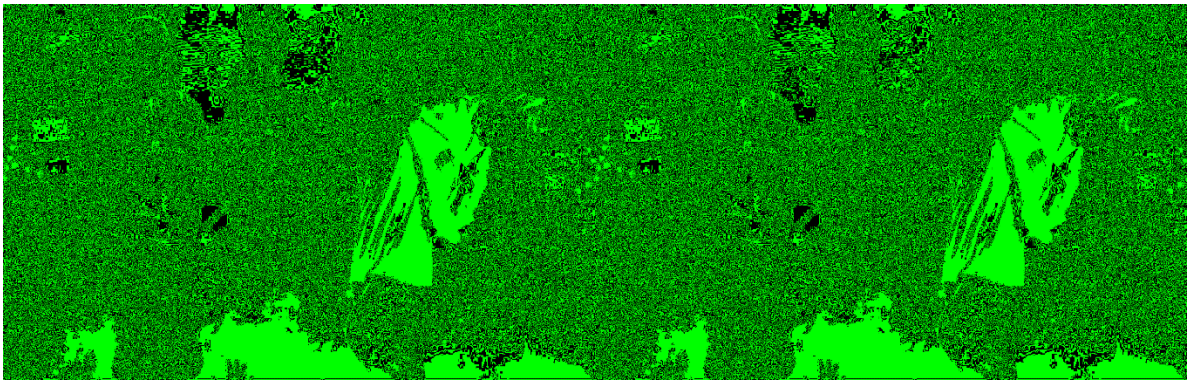
Warstwa czerwona, trzeci najmniej znaczący bit



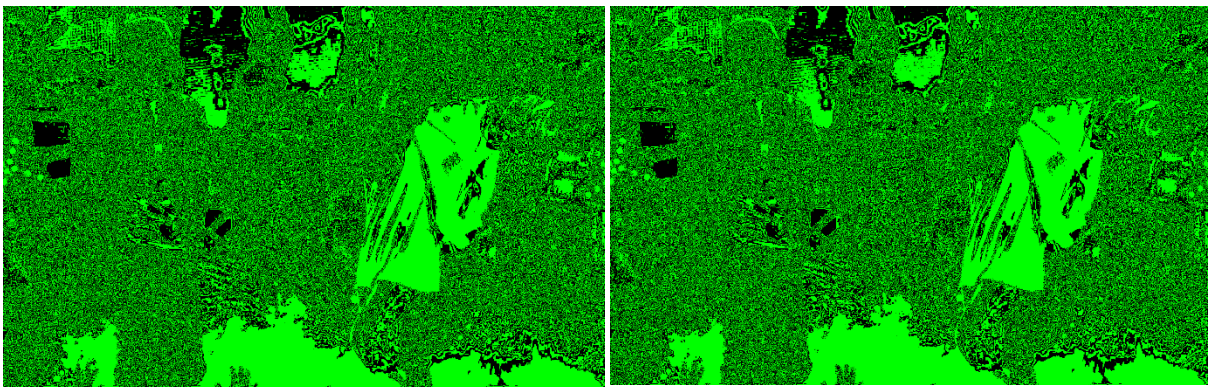
Warstwa zielona, pierwszy najmniej znaczący bit



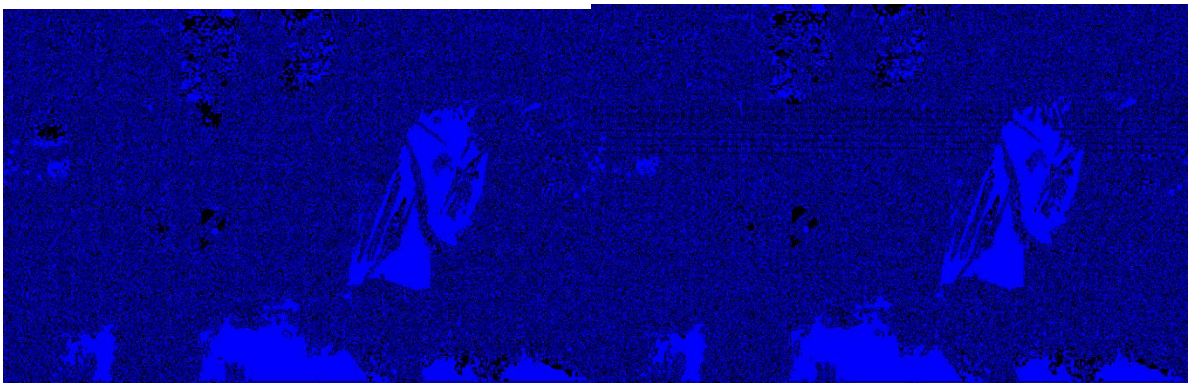
Warstwa zielona, drugi najmniej znaczący bit



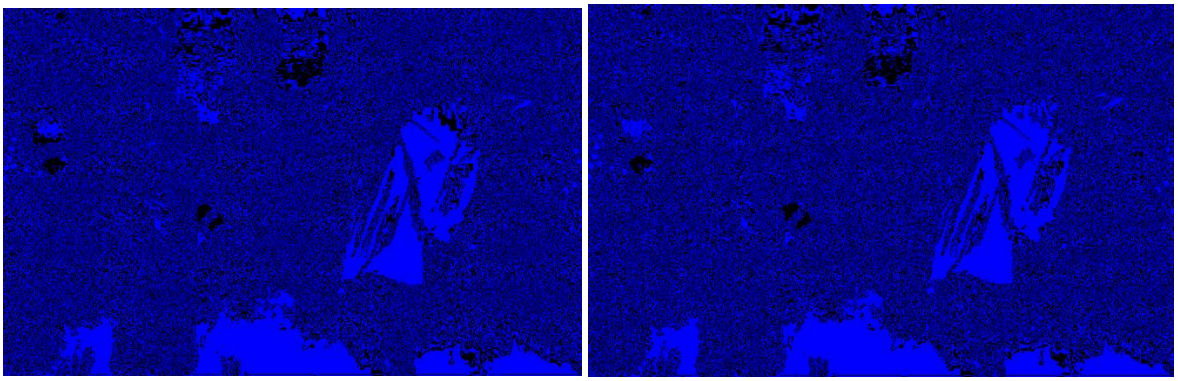
Warstwa zielona, trzeci najmniej znaczący bit



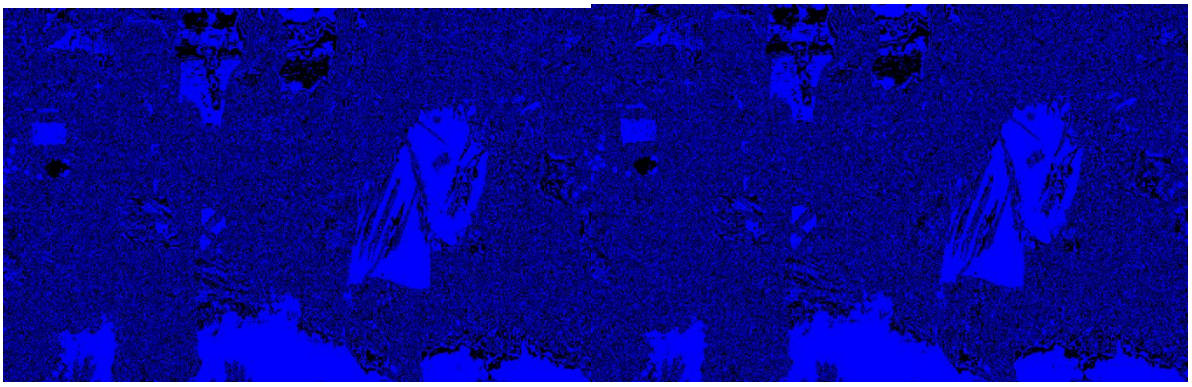
Warstwa niebieska, pierwszy najmniej znaczący bit



Warstwa niebieska, drugi najmniej znaczący bit



Warstwa niebieska, trzeci najmniej znaczący bit



Opis rozwiązania, komentarz dotyczący wyników

Obraz drugi, zapisany w kodzie pod zmienną `image_2` i będący reprezentacją zdjęcia załączonego do rozwiązania o nazwie `'cat_eye.jpg'` zostaje ukryty w obrazie pierwszym, zapisanym w kodzie pod zmienną `image_1` i będącym reprezentacją zdjęcia załączonego do rozwiązania o nazwie `'eminem-rihanna.jpg'`.

Przyjąłem sekwencyjny sposób ukrywania obrazu, najpierw iteruję po kolejnych pixelach w poziomie, gdy przeiterowane zostaną wszystkie pixele w wierszu, zmienna row zostaje zwiększona o 1 i iteracja zaczyna się po raz kolejny od lewej strony w nowym wierszu. Zapewnione jest to poprzez warunek `if(col>width_1)`.

Podobne operacje wykonywane są dla 3 kolejnych warstw RGB, natomiast należy zauważyć, że inicjalizacja zmiennych row i col następuje jeszcze przed zdefiniowaniem pętli dla warstw. Dzięki temu każda z warstw kolejnych w pętli zaczyna iterować przez pixele zdjęcia zaczynając w miejscu, w którym skończyła pętla dla warstwy poprzedniej.

Takie działanie pętli ukrywającej widać na załączonych powyżej wynikach, dla każdej z warstw (kolejno R G B) widać po kilka poziomych linii, co po pierwsze jest wskazówką na sekwencyjny sposób ukrywania danych, po drugie podpowiada, że iteracja następuje w poziomie, a po trzecie zauważyć można, że dla warstwy pierwszej – czerwonej – paski występują od samej góry, następnie dla warstw zielonej i niebieskiej następuje obniżenie pasków w dół.

Warto dodatkowo wspomnieć, że na pozostałych warstwach RGB dla drugich i trzecich najmniej znaczących bitów nie widać żadnych zmian. Jest to spowodowane działaniem algorytmu opartego o najmniej znaczący bit, następuje modyfikacja jedynie na poziomie pierwszego najmniej znaczącego bitu.

Poniżej zamieszczone zostało zdjęcie prezentujące algorytm ukrywania.

```
row = 1;
col = 1;
counter = 0;
for warstwa = 1 : 1 : 3
    for i = 1 : 1 : height_2
       for j = 1 : 1 : width_2
                number = image_2(i,j,1);
                bin_number = de2bi(number,8);
                    for l = 1 : 1 : 8
                        LSB = mod(double(image_1(row,col,warstwa)), 2);
                        temp = double(xor(LSB, bin_number(l)));
                        image_1(row,col,warstwa) = image_1(row,col,warstwa) + temp;
                        col = col + 1;
                        if(col > width_1)
                            col = 1;
                            row = row + 1;
                        end
                        counter = counter + 1;
                    end

         end
    end
end
```