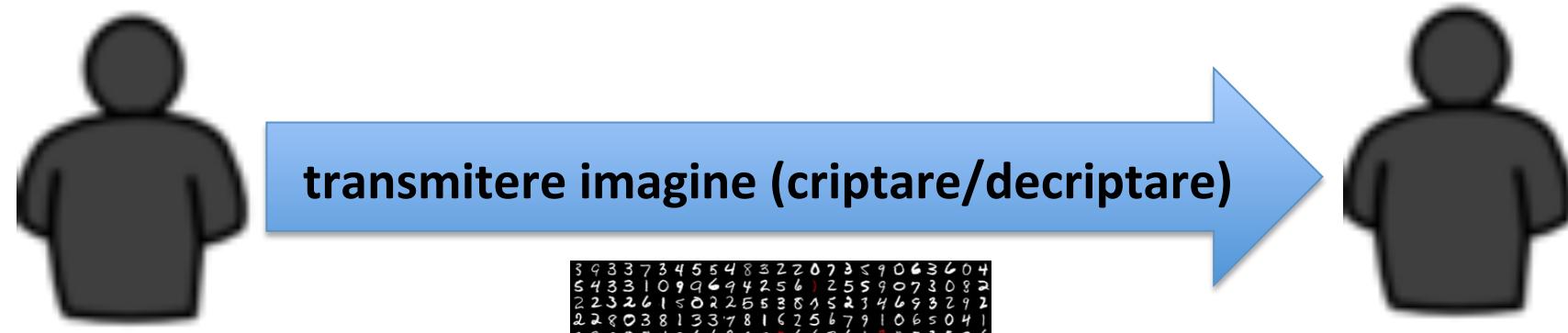


Ideea proiectului

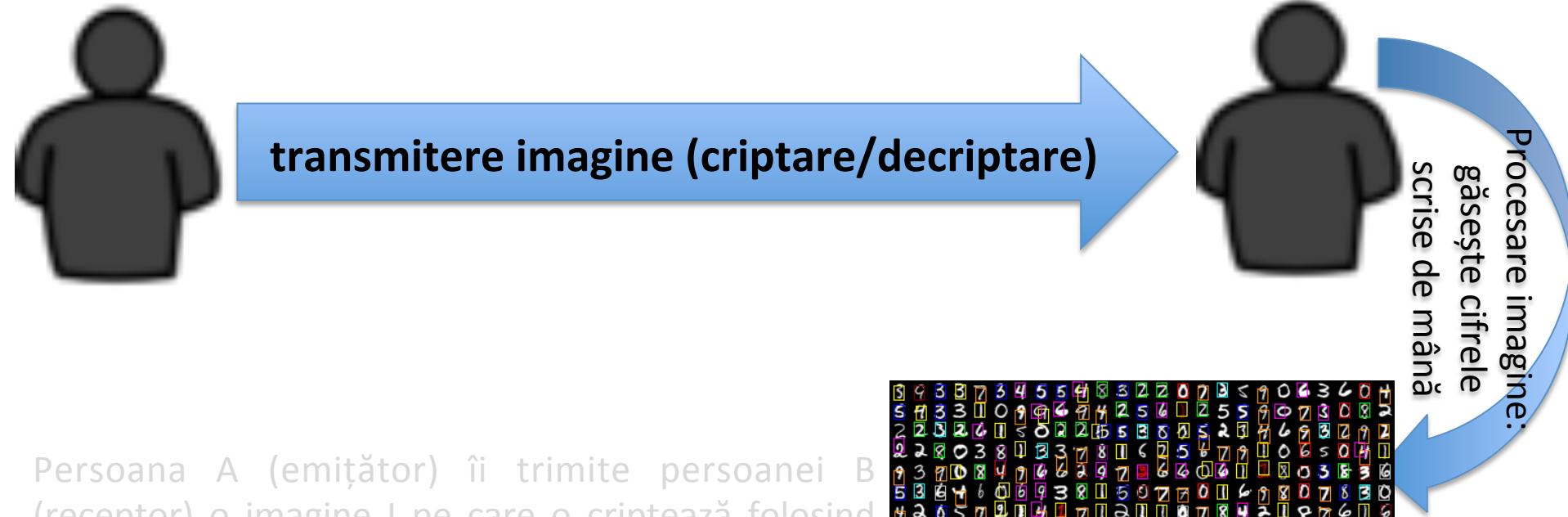


```
3 9 3 3 7 3 4 5 5 4 8 3 2 2 0 7 3 5 9 0 6 3 6 0 4  
5 4 3 3 1 0 9 9 6 9 4 2 5 6 1 2 5 5 9 0 7 3 0 8 2  
2 2 3 2 6 1 5 0 2 2 5 5 3 8 9 5 2 3 4 6 9 3 2 9 2  
2 2 8 0 3 8 1 3 3 7 8 1 6 2 5 6 7 9 1 0 6 5 0 4 1  
9 3 9 0 8 4 9 6 6 2 9 7 3 6 6 0 6 1 2 8 0 3 5 3 6  
5 3 6 4 6 0 6 9 3 8 1 5 0 7 7 0 1 6 9 8 0 7 8 3 0  
4 2 6 5 7 2 1 4 1 7 1 2 1 1 0 7 8 4 2 1 8 7 6 1 6  
3 8 1 2 0 4 4 8 0 9 0 9 3 7 6 1 8 4 1 1 8 9 4 0 2  
1 4 2 7 4 6 3 0 4 2 0 4 6 9 2 1 6 9 5 1 2 2 8 3 5  
4 8 8 8 2 1 7 7 3 6 6 7 7 4 0 1 0 4 5 5 7 0 8 0 1  
0 0 7 2 6 1 9 2 7 4 1 2 3 6 7 8 0 3 7 1 5 0 4 3 8  
3 6 4 0 6 0 5 5 1 5 8 7 8 4 7 3 9 1 2 3 2 9 5 2 9  
0 7 5 4 2 9 8 5 8 0 6 9 7 5 8 7 1 8 5 4 7 3 0 9 2  
5 9 7 9 5 1 6 7 2 6 4 2 6 8 3 8 9 3 9 5 4 9 9 7 9 7  
9 9 8 4 4 8 2 3 7 4 8 8 8 1 2 5 5 7 8 7 1 2 0 4 8  
7 6 9 3 5 7 5 7 6 7 4 0 2 2 1 3 1 4 2 8 9 7 7 1 5  
8 3 3 8 9 6 2 0 0 3 8 6 1 6 9 9 7 6 2 5 3 1 1 5 5  
9 5 7 0 4 3 0 9 9 3 5 5 2 4 6 5 0 4 2 3 3 4 5 0 5  
1 2 2 3 9 1 7 9 8 5 5 4 9 0 7 0 3 5 0 7 2 7 1 2 0  
8 8 6 1 1 6 6 4 4 0 8 5 8 2 2 4 3 3 9 4 2 6 2 9 5
```

Persoana A (emisator) îi trimit persoanei B (receptor) o imagine I pe care o criptează folosind un algoritm de criptare. B poate decripta imaginea criptată primită de la A obținând astfel imaginea inițială I.

3933734554832207359063604
5433109969425612559073082
2232615022553815234693292
2280381337816256791065041
9390849662973660611803536
5364606938150770169807830
4265721417121107842187616
3812044809093767841189402
1427463042046921695122835
4888217736677701045570801
0072619274123678037150438
3640605515878473912229529
0754298580699587185973092
5979516726268389396499797
9984482274888125578712048
769357576740221342897715
8338962003961699262531155
9570430993552465042344505
17739179855490351725120

Ideea proiectului



Persoana A (emisator) îi trimite persoanei B (receptor) o imagine I pe care o cripteză folosind un algoritm de criptare. B poate decripta imaginea criptată primită de la A obținând astfel imaginea inițială I după care procesează imaginea (folosește un algoritm de recunoaștere de cifre scrise de mână în imaginea I).





Ideea proiectului

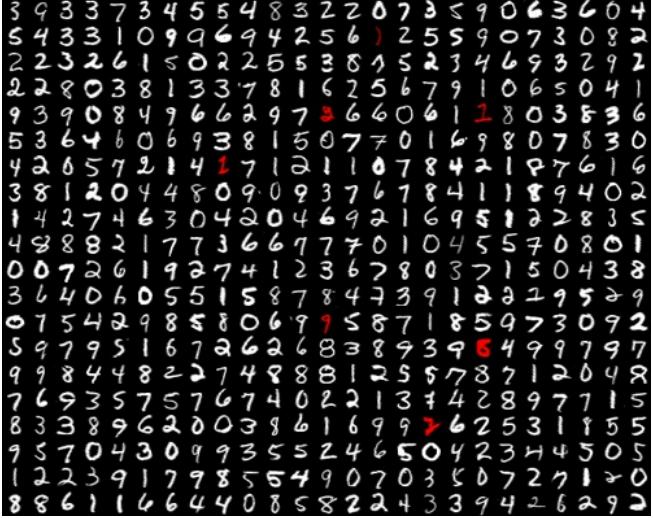


Persoana A (emisator) îi trimit persoanei B (receptor) o imagine I pe care o criptează folosind un algoritm de criptare. B poate decripta imaginea criptată primită de la A obținând astfel imaginea inițială I după care procesează imaginea (folosește un algoritm de recunoaștere de cifre scrise de mână în imaginea I). Persoana B trimit apoi persoanei A imaginea rezultată obținută pe care o criptează. Persoana A decriptează imaginea criptată primită de la B și poate vizualiza soluția, recunoașterea cifrelor scrise de mână.

Formatul BMP (bitmap)

Formatul BMP (bitmap)

- ❑ o imagine este un fișier binar
- ❑ imagine color RGB
- ❑ fiecare pixel are o culoare dată de un triplet (**R,G,B**) cu valori pe fiecare canal intre 0 si 255 = 1 octet/canal
- ❑ albastru = **(0,0 ,255)**, negru = **(0,0,0)**, rosu = **(255, 0, 0)**, verde = **(0,255,0)**
- ❑ 1 pixel color = 3 canale = 3 octeți
- ❑ dimensiuni 500 x 400 pixeli (lățime x înăltime)
- ❑ $500 \times 400 \times 3$ octeți = 600000 octeți



A 2D grid of numerical values representing pixel data for a BMP image. The values range from 0 to 255 and are arranged in a 400x500 grid. Some values are highlighted in red, such as '1' at [180, 353] and '2' at [204, 505]. The grid is enclosed in a black border.

3	9	3	3	7	3	4	5	5	4	8	3	2	2	0	7	3	<	9	0	6	3	6	0	4
5	4	3	3	1	0	9	9	6	9	4	2	5	6)	2	5	5	9	0	7	3	0	8	2
2	2	3	2	6	1	5	0	2	2	5	5	3	8	1	5	2	3	4	6	9	3	2	9	2
2	2	8	0	3	8	1	3	3	7	8	1	6	2	5	6	7	9	1	0	6	5	0	4	1
9	3	9	0	8	4	9	6	6	2	9	7	3	6	6	0	6	1	1	8	0	3	5	3	6
5	3	6	4	6	0	6	9	3	8	1	5	0	7	7	0	1	6	9	8	0	7	8	3	0
4	2	6	5	7	2	1	4	1	7	1	2	1	1	0	7	8	4	2	1	8	7	6	1	6
3	8	1	2	0	4	4	8	0	9	0	9	3	7	6	1	8	4	1	1	8	9	4	0	2
1	4	2	7	4	6	3	0	4	2	0	4	6	9	2	1	6	9	5	1	2	2	8	3	5
4	8	8	8	2	1	7	7	3	6	6	7	7	7	0	1	0	4	5	5	7	0	8	0	1
0	0	7	2	6	1	9	2	7	4	1	2	3	6	7	8	0	3	7	1	5	0	4	3	8
3	6	4	0	6	0	5	5	1	5	8	7	8	4	7	3	9	1	2	2	1	9	5	2	9
0	7	5	4	2	9	8	5	8	0	6	9	7	5	8	7	1	8	5	9	7	3	0	9	2
5	9	7	9	5	1	6	7	2	6	2	6	8	3	8	9	3	9	5	4	9	9	7	9	7
9	9	8	4	4	8	2	2	7	4	8	8	8	1	2	5	5	7	8	7	1	2	0	4	8
7	6	9	3	5	7	5	7	6	7	4	0	2	2	1	3	4	2	8	9	7	1	1	5	
8	3	3	8	9	6	2	0	0	3	8	6	1	6	9	9	7	6	2	5	3	1	1	5	5
9	5	7	0	4	3	0	9	9	3	5	5	2	4	6	5	0	4	2	3	4	4	5	0	5
1	2	2	3	9	1	7	9	8	5	5	4	9	0	7	0	3	5	0	7	2	7	1	2	0
8	8	6	1	1	4	6	4	4	0	8	5	8	2	2	4	3	3	9	4	2	6	2	9	2

test.bmp

Formatul BMP (bitmap)

 **test.bmp** 600 KB
Modified: Saturday, 24 November 2018 at 15:48
[Add Tags...](#)

General:

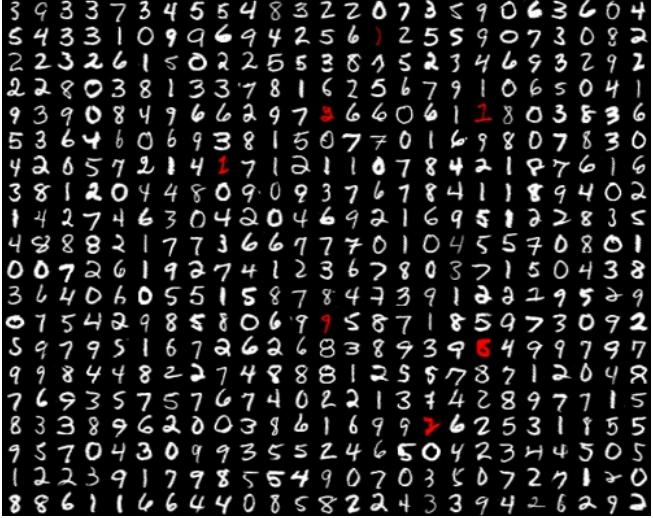
Kind: Windows bitmap image
Size: 600'054 bytes (602 KB on disk)
Where: Macintosh HD ▸ Users ▸ bogdan ▸ FMI ▸ PP ▸ Bogdan ▸ 2018-2019_Proiect ▸ projectPP ▸ date ▸ recunoasterePatternuri
Created: Thursday, 22 November 2018 at 00:04
Modified: Saturday, 24 November 2018 at 15:48
 Stationery pad
 Locked

More Info:

Dimensions: 500 × 400
Color space: RGB
Alpha channel: No

Name & Extension:

test.bmp



A grid of binary digits (0s and 1s) representing the content of the BMP file. The digits are arranged in a 25x25 grid, corresponding to the 500x400 pixel dimensions of the image. A red oval highlights the first few lines of the grid, starting from the top-left corner.

3	9	3	3	7	3	4	5	5	4	8	3	2	2	0	7	3	<	9	0	6	3	6	0	4
5	4	3	3	1	0	9	9	6	9	4	2	5	6)	2	5	5	9	0	7	3	0	8	2
2	2	3	2	6	1	5	0	2	2	5	5	3	8	1	5	2	3	4	6	9	3	2	9	2
2	2	8	0	3	8	1	3	3	7	8	1	6	2	5	6	7	9	1	0	6	5	0	4	1
9	3	9	0	8	4	9	6	6	2	9	7	3	6	6	0	6	1	2	8	0	3	5	3	6
5	3	6	4	6	0	6	9	3	8	1	5	0	7	7	0	1	6	9	8	0	7	8	3	0
4	2	6	5	7	2	1	4	1	7	1	2	1	1	0	7	8	4	2	1	8	7	6	1	6
3	8	1	2	0	4	4	8	0	9	0	9	3	7	6	1	8	4	1	1	8	9	4	0	2
1	4	2	7	4	6	3	0	4	2	0	4	6	9	2	1	6	9	5	1	2	2	8	3	5
4	8	8	8	2	1	7	7	3	6	6	7	7	7	0	1	0	4	5	5	7	0	8	0	1
0	0	7	2	6	1	9	2	7	4	1	2	3	6	7	8	0	3	7	1	5	0	4	3	8
3	6	4	0	6	0	5	5	1	5	8	7	8	4	7	3	9	1	2	2	1	9	5	2	9
0	7	5	4	2	9	8	5	8	0	6	9	7	5	8	7	1	8	5	9	7	3	0	9	2
5	9	7	9	5	1	6	7	2	6	2	6	8	3	8	9	3	9	5	4	9	9	7	9	7
9	9	8	4	4	8	2	2	7	4	8	8	8	1	2	5	5	7	8	7	1	2	0	4	8
7	6	9	3	5	7	5	7	6	7	4	0	2	2	1	3	4	2	8	9	7	7	1	5	5
8	3	3	8	9	6	2	0	0	3	8	6	1	6	9	9	7	6	2	5	3	1	1	5	5
9	5	7	0	4	3	0	9	9	3	5	5	2	4	6	5	0	4	2	3	4	5	0	5	5
1	2	2	3	9	1	7	9	8	5	5	4	9	0	7	0	3	5	0	7	2	7	1	2	0
8	8	6	1	1	4	6	4	4	0	8	5	8	2	2	4	3	3	9	4	2	6	2	9	2

test.bmp

$500 \times 400 \times 3 \text{ octet}i = 600000 \text{ octet}i + 54 \text{ octet}i \text{ (header)} = 600054 \text{ octet}i$

Formatul BMP (bitmap)

 **cifra0.bmp** 594 bytes
Modified: Wednesday, 22 November 2017 at 18:14
[Add Tags...](#)

General:

Kind: Windows bitmap image
Size: 594 bytes (4 KB on disk)
Where: Macintosh HD ▶ Users ▶ bogdan ▶ PMI ▶ PP ▶ Bogdan ▶ 2018-2019 ▶ Project ▶ projectPP ▶ date ▶ recunoasterePatternuri
Created: Saturday, 18 November 2017 at 13:08
Modified: Wednesday, 22 November 2017 at 18:14
 Stationery pad
 Locked

More Info:

Dimensions: 11 × 15
Color space: RGB
Alpha channel: No

Name & Extension:

cifra0.bmp



cifra0.bmp

- $11 \times 15 \times 3 \text{ octeți} = 495 \text{ octeți} + 54 \text{ octeți (header)} = 549 \text{ octeți}$
Se adaugă octeți de padding astfel încât fiecare linie e multiplu de 4

Formatul BMP (bitmap)

 **cifra0.bmp** 594 bytes
Modified: Wednesday, 22 November 2017 at 18:14

Add Tags...

▼ General:

Kind: Windows bitmap image
Size: 594 bytes (4 KB on disk)
Where: Macintosh HD ▶ Users ▶ bogdan ▶ VMI ▶ PP ▶ Bogdan ▶ 2018-2019 ▶ Project ▶ projectPP ▶ date ▶ recunoasterePatternuri
Created: Saturday, 18 November 2017 at 13:08
Modified: Wednesday, 22 November 2017 at 18:14
 Stationery pad
 Locked

▼ More Info:

Dimensions: 11 × 15
Color space: RGB
Alpha channel: No

▼ Name & Extension:

cifra0.bmp



cifra0.bmp

$(11 \times 3 + 3) \times 15 = 540$ octeți + 54 octeți (header) = 594 octeți
Se adaugă 3 octeți de padding (de obicei au valoarea 0)

Formatul BMP (bitmap)

- formatul BMP cuprinde o zonă de date cu dimensiune fixă, numită *header* și o zonă de date cu dimensiune variabilă care conține pixelii imaginii propriu-zise.
- header-ul, care ocupă primii 54 de octeți ai fișierului, conține informații despre formatul BMP, precum și informații despre dimensiunea imaginii (octetul cu numărul 2), numărul de octeți utilizați pentru reprezentarea unui pixel etc.
- dimensiunea imaginii în pixeli este exprimată sub forma $W \times H$ (lățime × înăltime). Lățimea imaginii exprimată în pixeli este memorată pe patru octeți fără semn începând cu octetul al 18-lea din header, iar înăltimea este memorată pe următorii 4 octeți fără semn, respectiv începând cu octetul al 22-lea din header.

Modulul de criptare/decriptare

CRİPTAREA İMAGİNİLƏR



CRİPTARE



DECRYPTARE



CIFRU SIMETRIC

Un **cifru simetric** este format din:

- un algoritm de criptare
- un algoritm de decriptare
- o cheie secretă comună



NOȚIUNI PRELIMINARE

- Un **generator de numere pseudo-aleatoare** este un algoritm determinist care generează o secvență de numere având proprietăți statistice asemănătoare celor ale unei secvențe de numere perfect aleatoare plecând de la o valoare inițială (seed).
- Generatorul **XORSHIFT32** propus de George Marsaglia în 2003 generează numere întregi fără semn pe 32 de biți, cu un caracter pseudo-aleator foarte bun, folosind operații de deplasare pe biți și XOR.

```
unsigned int r, seed, k;  
.....  
r = seed;  
for(k = 0; k < n; k++)  
{  
    r = r ^ r << 13;  
    r = r ^ r >> 17;  
    r = r ^ r << 5;  
    printf("%u\n", r);  
}
```

```
seed = 1000;  
266172694  
3204629577  
385443340  
2099747088  
1321081938  
733430728  
3121244111  
680290934  
.....
```

NOȚIUNI PRELIMINARE

- O **permutare aleatoare de lungime n** este o permutare selectată sau generată aleator din mulțimea celor $n!$ permutări de lungime n existente.
- O metodă eficientă de generare a unei permutări aleatoare p de lungime n este **algoritmul lui Durstenfeld** (1964):

```
unsigned int r, n, k, p[100];
.....
for(k = 0; k < n; k++)
    p[k] = k;

for(k = n-1; k >= 1; k--)
{
    r = random(0, k);
    aux = p[r];
    p[r] = p[k];
    p[k] = aux;
}
```

$n = 6$

k	r	p
-	-	1 2 3 4 5 6
5	3	1 2 3 6 5 4
4	1	1 5 3 6 2 4
3	1	1 6 3 5 2 4
2	2	1 6 3 5 2 4
1	0	6 1 3 5 2 4

unde prin `random(0, k)` am notat un număr aleator cuprins între 0 și k .

NOȚIUNI PRELIMINARE

- Fie $I = (I_{i,j})_{\substack{0 \leq i < H \\ 0 \leq j < W}}$ o imagine color cu lățimea de W pixeli și înălțimea de H pixeli în formă matriceală:

$$I = \begin{pmatrix} I_{0,0} & I_{0,1} & \dots & I_{0,W-1} \\ I_{1,0} & I_{1,1} & \dots & I_{1,W-1} \\ \dots & \dots & \dots & \dots \\ I_{H-1,0} & I_{H-1,1} & \dots & I_{H-1,W-1} \end{pmatrix}$$

Liniarizarea imaginii I presupune crearea unui tablou unidimensional L prin alipirea liniilor tabloului bidimensional I , de sus în jos:

$$L = \left(\underbrace{I_{0,0}, \dots, I_{0,W-1}}_{\text{Linia 0}}, \underbrace{I_{1,0}, \dots, I_{1,W-1}}_{\text{Linia 1}}, \dots, \underbrace{I_{H-1,0}, \dots, I_{H-1,W-1}}_{\text{Linia } H-1} \right) = (L_0, L_1, \dots, L_{W*H-1})$$

- Valoarea unui pixel L_k va fi un triplet de octeți fără semn $L_k = (L_k^R, L_k^G, L_k^B)$.
- Vom nota cu \oplus operația sau-exclusiv (XOR) între 2 octeți fără semn.
- Pentru pixelii $P_1 = (P_1^R, P_1^G, P_1^B)$ și $P_2 = (P_2^R, P_2^G, P_2^B)$ vom nota cu $P_1 \oplus P_2$ pixelul $(P_1^R \oplus P_2^R, P_1^G \oplus P_2^G, P_1^B \oplus P_2^B)$.
- Pentru un pixel $P = (P^R, P^G, P^B)$ și un întreg fără semn X pe 32 de biți format din octetii fără semn (X_3, X_2, X_1, X_0) vom nota cu $P \oplus X$ pixelul $(P^R \oplus X_2, P^G \oplus X_1, P^B \oplus X_0)$.

CIFRUL SIMETRIC PROPUȘ



Permutarea
pixelilor



Modificarea
valorilor
pixelilor



- **Cheia secretă comună** este compusă din valorile R_0 și SV , ambele numere întregi nenule fără semn pe 32 de biți.
- Pentru o imagine color în clar P de dimensiune $W \times H$ în formă liniară $P = (P_0, P_1, \dots, P_{W*H-1})$ și cheia secretă (R_0, SV) **algoritmul de criptare** este următorul:
 - a) se generează o secvență $R = (R_1, R_2, \dots, R_{2*W*H-1})$ de numere întregi aleatoare fără semn pe 32 de biți, folosind generatorul XORSSHIFT32 initializat cu o valoare nenulă R_0 ;
 - b) se generează o permutare aleatoare σ de dimensiune $W \times H$, folosind algoritmul lui Durstenfeld și numerele pseudo-aleatoare R_1, \dots, R_{W*H-1} ;
 - c) se permute pixelii imaginii P conform permutării σ , obținându-se o imagine intermediară P' , respectiv $P'_{\sigma(k)} = P_k$ pentru orice $k \in \{0, 1, \dots, W * H - 1\}$;

CIFRUL SIMETRIC PROPUŞ

d) imaginea criptată $C = (C_0, C_1, \dots, C_{W*H-1})$ se obține aplicând asupra fiecărui pixel al imaginii $P' = (P'_0, P'_1, \dots, P'_{W*H-1})$ următoarea relație de substituție:

$$C_k = \begin{cases} SV \oplus P'_0 \oplus R_{W*H}, & k = 0 \\ C_{k-1} \oplus P'_k \oplus R_{W*H+k}, & k \in \{1, 2, \dots, W * H - 1\} \end{cases}$$

- Cifrul fiind unul simetric, **algoritmul de decriptare** este complementar celui de criptare.
- Astfel, algoritmul de decriptare al unei imagini color criptate C de dimensiune $W \times H$ în formă liniară $C = (C_0, C_1, \dots, C_{W*H-1})$ și cheia secretă (R_0, SV) este următorul:
 - a) se generează o secvență $R = (R_1, R_2, \dots, R_{2*W*H-1})$ de numere întregi aleatoare fără semn pe 32 de biți, folosind generatorul XORSHIFT32 initializat cu valoarea nenulă R_0 ;

CIFRUL SIMETRIC PROPUȘ

- b) se generează permutarea aleatoare σ de dimensiune $W \times H$, folosind algoritmul lui Durstenfeld și numerele pseudo-aleatoare R_1, \dots, R_{W*H-1} , după care se calculează inversa sa σ^{-1} ;
- c) se aplică asupra fiecărui pixel din imaginea criptată $C = (C_0, \dots, C_{W*H-1})$ inversa relației de substituție folosită în procesul de criptare, obținându-se o imagine intermediară $C' = (C'_0, C'_1, \dots, C'_{W*H-1})$:

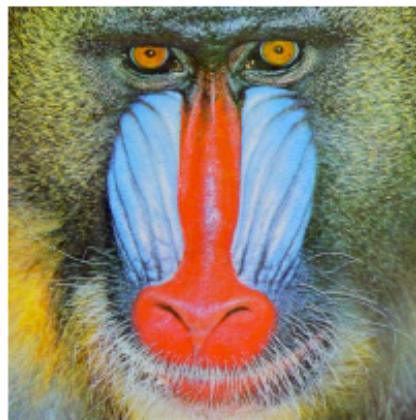
$$C'_k = \begin{cases} SV \oplus C_0 \oplus R_{W*H}, & k = 0 \\ C_{k-1} \oplus C_k \oplus R_{W*H+k}, & k \in \{1, 2, \dots, W * H - 1\} \end{cases}$$

- d) imaginea decriptată $D = (D_0, D_1, \dots, D_{W*H-1})$ se obține permutând pixelii imaginii C' conform permutării σ^{-1} , respectiv $D_{\sigma^{-1}(k)} = C'_k$ pentru orice $k \in \{0, 1, \dots, W * H - 1\}$.

ANALIZA PERFORMANȚELOR CIFRULUI PROPUȘ

- Asupra unei imagini criptate, o persoană neautorizată (un atacator care nu cunoaște cheia secretă) poate efectua mai multe tipuri de atacuri pentru a afla fie cheia secretă, fie imaginea în clar (în multe cazuri, chiar și determinarea doar a unei părți dintr-o imagine poate furniza informații foarte importante atacatorului).
- Unul dintre cele mai simple atacuri îl constituie **atacul în frecvență**, prin care atacatorul mai întâi va determina, pe fiecare canal de culoare, frecvențele valorilor pixelilor, după care va încerca să asocieze valorile lor, în ordinea descrescătoare a frecvențelor, cu valorile pe care el le consideră ca fiind cele mai probabile.
- Pentru a rezista în fața unor astfel de atacuri de tip statistic, un cifru trebuie să producă, indiferent de cheia secretă utilizată, imagini criptate cu o distribuție uniformă a valorilor pixelilor din fiecare canal de culoare, pentru a ascunde astfel distribuția neuniformă a valorilor pixelilor din imaginea inițială.
- Instrumentul de analiză vizuală cel mai des utilizat pentru a studia distribuția valorilor pixelilor unei imagini color îl constituie *histograma culorilor*, în care sunt reprezentate grafic frecvențele valorilor pixelilor unei imagini, separat pentru fiecare canal de culoare.

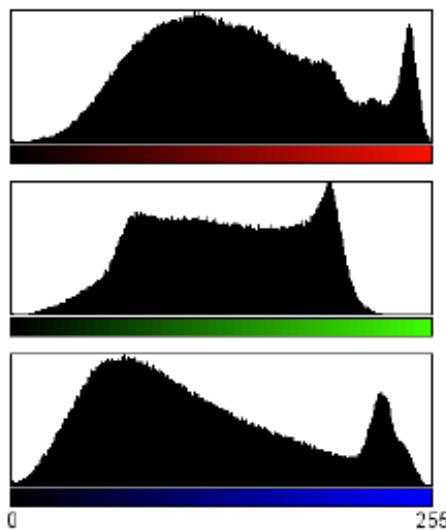
ANALIZA PERFORMANȚELOR CIFRULUI PROPUȘ



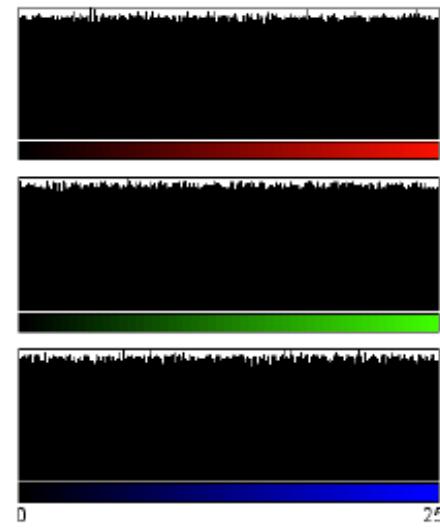
Imaginea în clar



Imaginea criptată



Histograma imaginii în clar



Histograma imaginii criptate

ANALIZA PERFORMANȚELOR CIFRULUI PROPUȘ

- Analiza vizuală a uniformității distribuției valorilor pixelilor folosind histograma culorilor prezintă mai multe dezavantaje, unul dintre ele fiind faptul că precizia evaluării depinde într-un mod decisiv de acuitatea vizuală a evaluatorului uman și de calitatea grafică a histogramei.
- Dezavantajele analizei vizuale pot fi eliminate prin utilizarea unui instrument statistic de evaluare a uniformității distribuției valorilor dintr-un sir, respectiv testul χ^2 (**testul chi-pătrat**).
- Valoarea testului χ^2 corespunzătoare unei imagini de dimensiune $m \times n$ pixeli se calculează folosind formula următoare:

$$\chi^2 = \sum_{i=0}^{255} \frac{(f_i - \bar{f})^2}{\bar{f}}$$

unde f_i este frecvența valorii i ($0 \leq i \leq 255$) pe un canal de culoare al imaginii, iar \bar{f} este frecvența estimată teoretic a oricărei valori i , respectiv $\bar{f} = \frac{m \times n}{256}$.

ANALIZA PERFORMANȚELOR CIFRULUI PROPUȘ

- O valoare a testului mai mică sau egală cu valoarea teoretică de prag $\chi_0^2 = 293.25$ indică o histogramă uniformă pe canalul de culoare respectiv, în timp ce o valoare mai mare indică o histogramă neuniformă.
- Pentru cele două imagini de mai sus, valorile testului χ^2 pe canalele de culoare RGB sunt următoarele:
 - pentru imaginea în clar: (211104.79, 348462.19, 197839.49)
 - pentru imaginea criptată: (264.48, 251.41, 330.49)

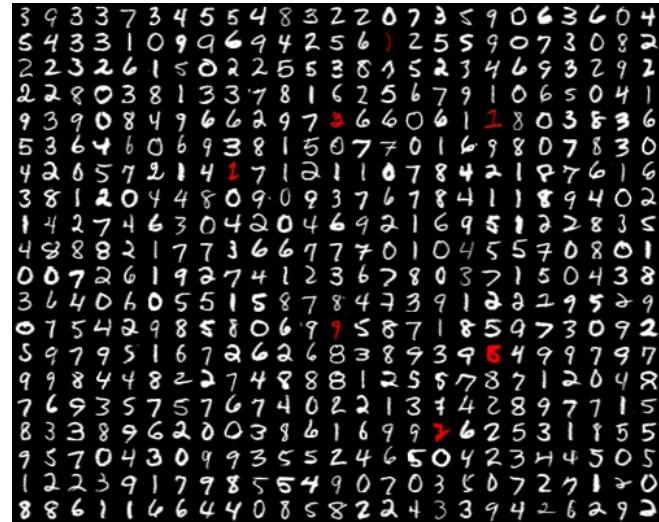
- 1) Implementați generatorul de numere pseudo-aleatoare XORSHIFT32 sub forma unei funcții sau a unei macrodefiniții.
- 2) Scrieți o funcție care să încarce o imagine BMP în memoria internă în formă liniarizată. Funcția va avea ca parametru calea imaginii BMP.
- 3) Scrieți o funcție care să salveze în memoria externă o imagine BMP stocată în formă liniarizată în memoria internă. Funcția va avea ca parametru calea imaginii BMP.
- 4) Scrieți o funcție care să cripteze o imagine BMP folosind algoritmul de criptare prezentat. Funcția va avea ca parametri calea imaginii inițiale, calea imaginii criptate și calea unui fișier text care conține cheia secretă.
- 5) Scrieți o funcție care să decripteze o imagine BMP criptată folosind algoritmul de decriptare prezentat. Funcția va avea ca parametri calea imaginii inițiale, calea imaginii criptate și calea unui fișier text care conține cheia secretă.

- 6) Scrieți o funcție care să afișeze valorile testului χ^2 pentru fiecare canal de culoare al unei imagini BMP. Funcția va avea ca parametru calea imaginii.
- 7) Scrieți un program care să realizeze următoarele operații:
 - criptarea unei imagini color BMP și salvarea imaginii criptate în memoria externă (căile ambelor imagini și a fișierului text care conține cheia secretă se vor citi de la tastatură);
 - decriptarea unei imagini color BMP criptate și salvarea imaginii decriptate în memoria externă (căile ambelor imagini și a fișierului text care conține cheia secretă se vor citi de la tastatură);
 - afișarea pe ecran a valorilor testului χ^2 pentru imaginea inițială și imaginii criptată, pe fiecare canal de culoare.

Modulul de recunoaștere de
pattern-uri (cifre scrise de mâňă)

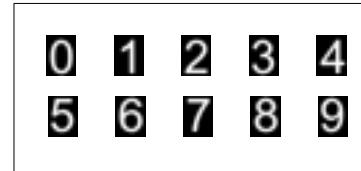
Template matching

- metoda de *template matching* constă în găsirea de părți mici ale unei imagini care se potrivesc cu o imagine şablon.



test.bmp

- pentru problema recunoașterii cifrelor scrise de mâna vom folosi pe post de şabloane imagini cu 10 cifre.



cifra0.bmp

cifra1.bmp

...

cifra9.bmp

Template matching

Algoritmul de template matching aplicat pentru o imagine color I și un şablon S (o imagine color ce reprezintă o cifră) funcționează astfel:

- se transformă imaginea color I și şablonul S în imagini grayscale. Un punct de pornire este sursa *grayscale.c* din arhivă;
- se glisează şablonul curent pe imaginea I centrând şablonul în fiecare punct al imaginii I. Se calculează pentru fiecare poziție (x, y) din imaginea I corelația (<https://en.wikipedia.org/wiki/Cross-correlation>) dintre şablonul curent și conținutul corespunzător al imaginii dat de fereastra $f_I = f_I(x, y)$ centrată în punctul (x, y) . Formula de calcul a corelației dintre şablonul S și fereastra f_I este următoarea:

3	9	3	3	7	3	4	5	5	4	8	3	2	2	0	7	3	5	9	0	6	3	6	0	4	
5	4	3	3	1	0	9	9	6	9	4	2	5	6	1	2	5	5	9	0	7	3	0	8	2	
2	2	3	2	6	1	5	0	2	2	5	5	3	8	1	5	2	3	4	6	9	3	2	9	2	
2	2	8	0	3	8	1	3	3	7	8	1	6	2	5	6	7	9	1	0	6	5	0	4	1	
9	3	9	0	8	4	9	6	6	2	9	7	3	6	4	0	6	1	1	8	0	3	5	3	6	
5	3	6	4	6	0	6	9	3	8	1	5	0	7	7	0	1	6	9	8	0	7	8	3	0	
4	4	2	0	5	7	2	1	4	1	7	1	2	1	1	0	7	8	4	2	1	8	7	6	1	6
3	3	8	1	2	0	4	4	8	0	9	0	9	3	7	6	1	8	4	1	1	8	9	4	0	2
1	1	4	2	7	4	6	3	0	4	2	0	4	6	9	2	1	6	9	5	1	2	2	8	3	5
4	4	8	8	8	2	1	7	7	3	6	6	7	7	7	0	1	0	4	5	5	7	0	8	0	1
0	0	7	2	6	1	9	2	7	4	1	2	3	6	7	8	0	3	7	1	5	0	4	3	8	
3	3	6	4	0	6	0	5	5	1	5	8	7	8	4	7	3	9	1	2	3	2	9	5	2	9
0	0	7	5	4	2	9	8	5	8	0	6	9	9	5	8	7	1	8	5	9	7	3	0	9	2
5	5	9	7	9	5	1	6	7	2	6	2	6	8	3	8	9	3	9	6	4	9	9	7	9	7
9	9	8	4	4	8	2	2	7	4	8	8	8	1	2	5	5	7	8	7	1	2	0	4	8	
7	7	6	9	3	5	7	5	7	6	7	4	0	2	2	1	3	4	2	8	9	7	7	1	5	
8	8	3	3	8	9	6	2	0	0	3	8	6	1	6	9	9	7	6	2	5	3	1	1	5	5
9	9	5	7	0	4	3	0	9	9	3	5	5	2	4	6	5	0	4	2	3	4	5	0	5	
1	1	2	2	3	9	1	7	9	8	5	5	4	9	0	7	0	3	5	0	7	2	7	1	2	0
8	8	6	1	1	4	6	4	4	0	8	5	8	2	2	4	3	3	9	4	2	6	2	9	2	

0	1	2	3	4
5	6	7	8	9

Template matching

Algoritmul de template matching aplicat pentru o imagine color I și un şablon S (o imagine color ce reprezintă o cifră) funcționează astfel:

- se transformă imaginea color I și şablonul S în imagini grayscale. Un punct de pornire este sursa *grayscale.c* din arhivă;
- se glisează şablonul curent pe imaginea I centrând şablonul în fiecare punct al imaginii I. Se calculează pentru fiecare poziție (x, y) din imaginea I corelația (<https://en.wikipedia.org/wiki/Cross-correlation>) dintre şablonul curent și conținutul corespunzător al imaginii dat de fereastra $f_I = f_I(x, y)$ centrată în punctul (x, y) . Formula de calcul a corelației dintre şablonul S și fereastra f_I este următoarea:

0	9	3	3	7	3	4	5	5	4	8	3	2	2	0	7	3	5	9	0	6	3	6	0	4	
5	4	3	3	1	0	9	9	6	9	4	2	5	6	1	2	5	5	9	0	7	3	0	8	2	
2	2	3	2	6	1	5	0	2	2	5	5	3	8	1	5	2	3	4	6	9	3	2	9	2	
2	2	8	0	3	8	1	3	3	7	8	1	6	2	5	6	7	9	1	0	6	5	0	4	1	
9	3	9	0	8	4	9	6	6	2	9	7	3	6	4	0	6	1	1	8	0	3	5	3	6	
5	3	6	4	6	0	6	9	3	8	1	5	0	7	7	0	1	6	9	8	0	7	8	3	0	
4	4	2	0	5	7	2	1	4	1	7	1	2	1	1	0	7	8	4	2	1	8	7	6	1	6
3	3	8	1	2	0	4	4	8	0	9	0	9	3	7	6	1	8	4	1	1	8	9	4	0	2
1	1	4	2	7	4	6	3	0	4	2	0	4	6	9	2	1	6	9	5	1	2	2	8	3	5
4	4	8	8	2	1	7	7	3	6	6	7	7	7	0	1	0	4	5	5	7	0	8	0	1	
0	0	7	2	6	1	9	2	7	4	1	2	3	6	7	8	0	3	7	1	5	0	4	3	8	
3	3	6	4	0	6	0	5	5	1	5	8	7	8	4	7	3	9	1	2	2	1	9	5	2	9
0	0	7	5	4	2	9	8	5	8	0	6	9	9	5	8	7	1	8	5	9	7	3	0	9	2
5	5	9	7	9	5	1	6	7	2	6	2	6	8	3	8	9	3	9	6	4	9	9	7	9	7
9	9	8	4	4	8	2	2	7	4	8	8	8	1	2	5	5	7	8	7	1	2	0	4	8	
7	7	6	9	3	5	7	5	7	6	7	4	0	2	2	1	3	4	2	8	9	7	7	1	5	
8	8	3	3	8	9	6	2	0	0	3	8	6	1	6	9	9	7	6	2	5	3	1	1	5	5
9	9	5	7	0	4	3	0	9	9	3	5	5	2	4	6	5	0	4	2	3	4	5	0	5	
1	1	2	2	3	9	1	7	9	8	5	5	4	9	0	7	0	3	5	0	7	2	7	1	2	0
8	8	6	1	1	4	6	4	4	0	8	5	8	2	2	4	3	3	9	4	2	6	2	9	2	

0	1	2	3	4
5	6	7	8	9

Calculul corelatiei

Formula de calcul a corelației dintre şablonul S și fereastra f_I este următoarea:

$$\text{corr}(S, f_I) = \frac{1}{n} \sum_{(i,j) \in S} \frac{1}{\sigma_{f_I} \sigma_S} (f_I(i,j) - \bar{f}_I) (S(i,j) - \bar{S}), \text{ unde}$$

- n reprezintă numărul de pixeli în şablonul S (în particular pentru şabloanele utilizate de dimensiuni 11×15 pixeli avem $n = 11 * 15 = 165$);
- indicii i și j reprezintă linia i și coloana j în şablonul S (15 linii și 11 coloane);
- $S(i,j)$ reprezintă valoarea intensității grayscale a pixelului de la linia i și coloana j în şablonul S . Pentru o imagine grayscale, un pixel $P = (P^R, P^G, P^B)$ este reprezentat de un triplet cu toate valorile egale $P^R = P^G = P^B$. În acest caz valoarea intensității grayscale a lui P este P^R ;
- \bar{S} reprezintă media valorilor intensităților grayscale a pixelilor în fereastra S (media celor 165 de pixeli din şablonul S);
- σ_S reprezintă deviația standard a valorilor intensităților grayscale a pixelilor în şablonul S : $\sigma_S = \sqrt{\frac{1}{n-1} \sum_{(i,j) \in S} (S(i,j) - \bar{S})^2}$
- $f_I(i,j)$ reprezintă valoarea intensității grayscale a pixelului de la linia i și coloana j în fereastra f_I .
- \bar{f}_I reprezintă media valorilor intensităților grayscale a pixelilor din fereastra f_I (media celor 165 de pixeli din fereastra f_I);
- σ_{f_I} reprezintă deviația standard a valorilor intensităților grayscale a pixelilor în fereastra f_I : $\sigma_{f_I} = \sqrt{\frac{1}{n-1} \sum_{(i,j) \in f_I} (f_I(i,j) - \bar{f}_I)^2}$

Ferestre cu corelație mai mare decât un prag

Cifra 0, prag mic = 0.35 → multe detecții

3	0	3	3	7	3	4	5	5	4	8	3	2	2	0	7	3	< 9	0	3	0	0	0		
5	4	3	3	1	0	9	4	6	9	4	2	5	6	2	5	5	9	0	7	3	0	8	2	
2	2	3	2	6	1	5	0	2	2	5	5	3	8	0	5	2	3	5	4	9	3	2	9	
2	2	8	0	3	8	1	3	3	7	8	1	6	7	5	6	7	9	1	0	6	5	0	4	
9	3	9	0	8	4	9	6	4	2	9	7	3	6	0	6	1	1	8	0	3	5	3	6	
5	3	6	4	6	0	6	9	3	2	1	5	9	7	7	0	1	6	9	8	0	7	8	3	0
4	2	0	5	7	2	1	0	1	7	1	2	1	0	7	8	4	2	1	2	7	6	1	5	
3	8	1	2	0	4	4	8	0	0	0	3	7	6	7	8	4	1	1	8	9	4	0	2	
1	4	2	7	4	6	3	0	4	2	0	4	6	9	2	1	6	0	5	1	2	2	8	3	5
0	8	9	8	2	1	7	7	3	6	6	7	7	0	1	0	5	5	7	0	8	0	1	1	
0	0	7	2	6	1	9	2	7	4	1	0	3	6	2	8	0	3	7	1	5	0	4	3	8
3	6	0	0	6	0	5	5	1	5	8	7	8	4	2	3	9	1	3	8	1	2	9	5	2
0	7	5	4	9	9	8	5	8	0	6	9	7	5	8	7	1	8	5	0	7	3	0	9	2
5	9	7	9	5	1	6	7	6	4	2	6	8	3	8	9	3	9	8	4	9	9	7	9	0
9	9	8	4	4	8	2	0	7	4	8	8	1	2	5	6	7	8	7	1	2	0	4	8	
7	4	5	3	5	7	5	7	0	7	0	2	0	1	3	4	4	2	8	9	7	7	1	5	
8	3	3	8	9	6	2	0	0	3	8	6	1	6	9	7	6	2	5	3	1	1	5	5	
9	5	7	0	4	3	0	9	9	3	5	5	2	0	6	5	0	4	2	3	4	4	5	0	5
1	2	2	3	9	1	7	9	8	5	6	4	9	0	7	0	3	0	0	7	2	7	1	2	0
8	8	6	0	1	0	6	0	4	0	8	5	8	2	3	4	3	3	9	4	2	6	2	9	2

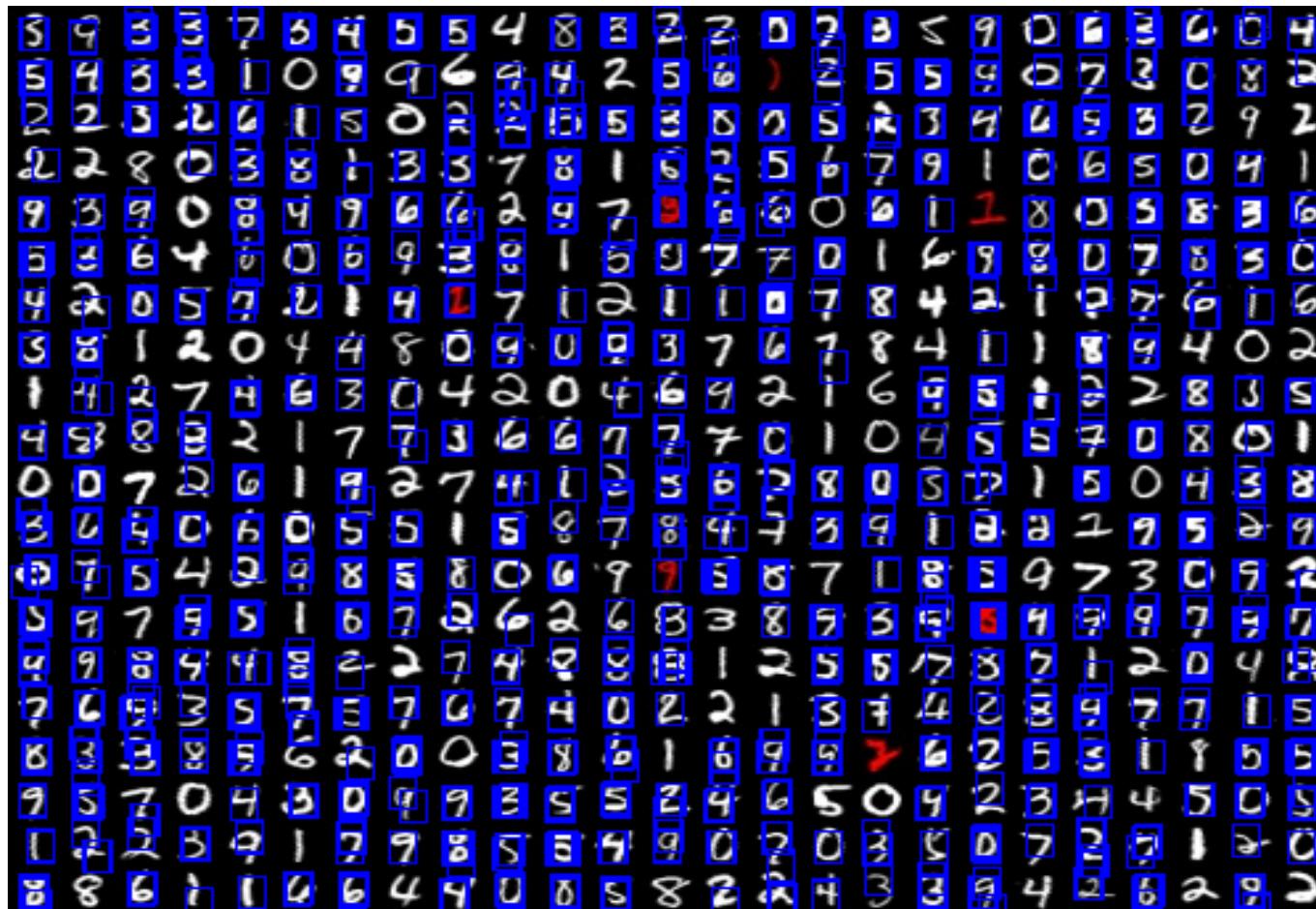
Ferestre cu corelație mai mare decât un prag

Cifra 0, prag mare = 0.70 → puține detecții

3	9	3	3	7	3	4	5	5	4	8	3	2	2	0	7	3	≤	9	0	6	3	6	0	4
5	4	3	3	1	0	9	9	6	9	4	2	5	6	1	2	5	5	9	0	7	3	0	8	2
2	2	3	2	6	1	5	0	2	2	5	5	3	8	1	5	2	3	4	6	9	3	2	9	2
2	2	8	0	3	8	1	3	3	7	8	1	6	2	5	6	7	9	1	0	6	5	0	4	1
9	3	9	0	8	4	9	6	6	2	9	7	3	6	6	0	6	1	1	8	0	3	8	3	6
5	3	6	4	6	0	6	9	3	8	1	5	0	7	7	0	1	6	9	8	0	7	8	3	0
4	2	0	5	7	2	1	4	1	7	1	2	1	1	0	7	8	4	2	1	8	7	6	1	6
3	8	1	2	0	4	4	8	0	9	0	9	3	7	6	7	8	4	1	1	8	9	4	0	2
1	4	2	7	4	6	3	0	4	2	0	4	6	9	2	1	6	9	5	1	2	2	8	3	5
4	8	8	8	2	1	7	7	3	6	6	7	7	0	1	0	4	5	5	7	0	8	0	1	
0	0	7	2	6	1	9	2	7	4	1	2	3	6	2	8	0	3	7	1	5	0	4	3	8
3	6	4	0	6	0	5	5	1	5	8	7	8	4	7	3	9	1	2	2	1	9	5	2	9
0	7	5	4	2	9	8	5	8	0	6	9	7	5	8	7	1	8	5	9	7	3	0	9	2
5	9	7	9	5	1	6	7	2	6	2	6	8	3	8	9	3	9	5	4	9	9	7	9	7
9	9	8	4	4	8	2	2	7	4	8	8	8	1	2	5	5	7	8	7	1	2	0	4	8
7	6	9	3	5	7	5	7	6	7	4	0	2	2	1	3	4	2	8	9	7	7	1	5	
8	3	3	8	9	6	2	0	3	8	6	1	6	9	9	2	6	2	5	3	1	1	5	5	
9	5	7	0	4	3	0	9	9	3	5	5	2	4	6	5	0	4	2	3	4	4	5	0	5
1	2	2	3	9	1	7	9	8	5	5	4	9	0	7	0	3	5	0	7	2	7	1	2	0
8	8	6	1	1	6	6	4	4	0	8	5	8	2	2	4	3	3	9	4	2	6	2	9	3

Ferestre cu corelație mai mare decât un prag

Cifra 5, prag mic = 0.35 → multe detecții



Ferestre cu corelație mai mare decât un prag

Cifra 5, prag mare = 0.70 → puține detecții

3	9	3	3	7	3	4	5	5	4	8	3	2	2	0	7	3	<	9	0	6	3	6	0	4
5	4	3	3	1	0	9	9	6	9	4	2	5	6)	2	5	5	9	0	7	3	0	8	2
2	2	3	2	6	1	5	0	2	2	5	5	3	8	1	5	2	3	4	6	9	3	2	9	2
2	2	8	0	3	8	1	3	3	7	8	1	6	2	5	6	7	9	1	0	6	5	0	4	1
9	3	9	0	8	4	9	6	6	2	9	7	3	6	6	0	6	1	1	8	0	3	8	3	6
5	3	6	4	6	0	6	9	3	8	1	5	0	7	7	0	1	6	9	8	0	7	8	3	0
4	2	0	5	7	2	1	4	1	7	1	2	1	1	0	7	8	4	2	1	8	7	6	1	6
3	8	1	2	0	4	4	8	0	9	0	9	3	7	6	7	8	4	1	1	8	9	4	0	2
1	4	2	7	4	6	3	0	4	2	0	4	6	9	2	1	6	9	5	1	2	2	8	3	5
4	8	8	8	2	1	7	7	3	6	6	7	7	0	1	0	4	5	5	7	0	8	0	1	
0	0	7	2	6	1	9	2	7	4	1	2	3	6	2	8	0	3	7	1	5	0	4	3	8
3	6	4	0	6	0	5	5	1	5	8	7	8	4	7	3	9	1	2	2	1	9	5	2	9
0	7	5	4	2	9	8	5	8	0	6	9	7	5	8	7	1	8	5	9	7	3	0	9	2
5	9	7	9	5	1	6	7	2	6	2	6	8	3	8	9	3	9	5	4	9	9	7	9	7
9	9	8	4	4	8	2	2	7	4	8	8	8	1	2	5	5	7	8	7	1	2	0	4	8
7	6	9	3	5	7	5	7	6	7	4	0	2	2	1	3	4	2	8	9	7	7	1	5	
8	3	3	8	9	6	2	0	3	8	6	1	6	9	9	2	6	2	5	3	1	1	5	5	
9	5	7	0	4	3	0	9	9	3	5	5	2	4	6	5	0	4	2	3	4	4	5	0	5
1	2	2	3	9	1	7	9	8	5	5	4	9	0	7	0	3	5	0	7	2	7	1	2	0
8	8	6	1	1	6	6	4	4	0	8	5	8	2	2	4	3	3	9	4	2	6	2	9	3

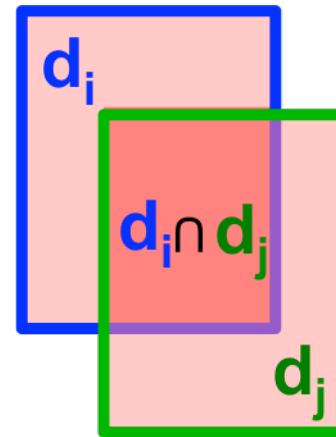
Ferestre cu corelație mai mare decât un prag

Cifrele 0-9, prag = 0.50 → multe detecții suprapuse



Ferestre cu corelație mai mare decât un prag

Cifrele 0-9, prag = 0.50 → multe detectii suprapuse

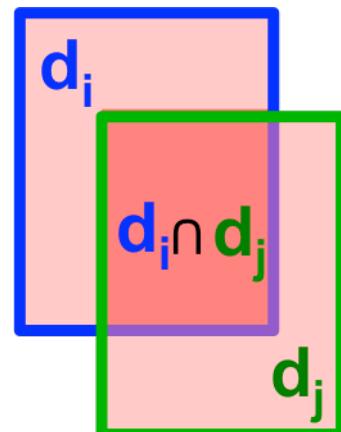


Pentru detectiile care se suprapun o păstrează pe cea cu scorul mai mare

Suprimarea non-maximelor

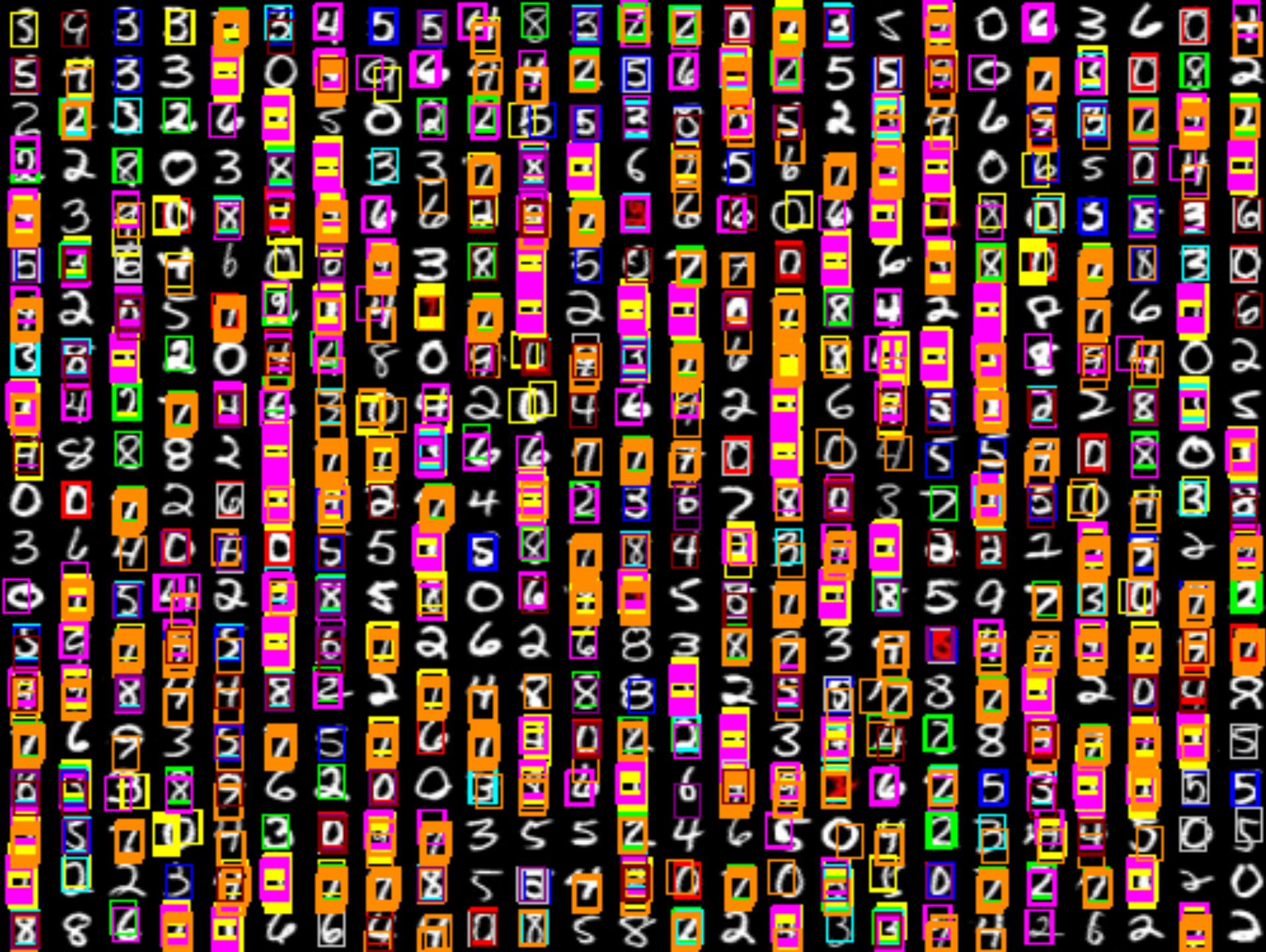
În practică, pentru evitarea acestui fenomen se folosește o tehnică numită *eliminarea non-maximelor*. Ea constă în următoarele:

- e) se pun într-un tablou unidimensional D toate detecțiile obținute pentru toate cele 10 şabloane;
- f) se sortează tabloul D descrescător în funcție de scorul de corelație al fiecărei detecții.
- g) se procesează tabloul D sortat de la stânga (deteceții cu scor foarte mare) la dreapta (deteceții cu scor foarte mic) astfel: toate detecțiile d_j care se suprapun spațial cu detecția curentă d_i , cu $i < j$ și deci și $\text{scor}(d_i) > \text{scor}(d_j)$ se elimină.



$$\text{suprapunere}(d_i, d_j) = \frac{\text{aria}(d_i \cap d_j)}{\text{aria}(d_i \cup d_j)} = \frac{\text{aria}(d_i \cap d_j)}{\text{aria}(d_i) + \text{aria}(d_j) - \text{aria}(d_i \cap d_j)}$$

În acest proiect vom considera că două ferestre se suprapun dacă suprapunerea lor spațială este > 0.2 .





Cerințe:

- 7) Scrieți o funcție care să implementeze operația de template matching dintre o imagine I și un şablon S . Funcția va avea ca parametri imaginea I , şablonul S și pragul p_S și furnizează f_I care au corelația mai mare decât pragul p_S . (1.5 puncte)
- 8) Scrieți o funcție care primește ca parametru o imagine I , o fereastră f_I și o culoare $C = (C^R, C^G, C^B)$ și desenează conturul ferestrei f_I în imaginea I folosind culoarea C (ca în exemplele de mai sus). (1 punct)
- 9) Folosind funcția `qsort` din librăria `stdlib.h` scrieți o funcție care sortează un tablou D de detecții în ordinea descrescătoare a corelațiilor detecțiilor. (0.25 puncte)
- 10) Scrieți o funcție care implementează algoritmul de eliminare a non-maximelor. (1.5 puncte)

11) Scrieți un program care să realizeze următoarele operații: (0.5 puncte)

- cripteză o imagine color BMP și salvează imaginea criptată în memoria externă (căile ambelor imaginii și a fișierului text care conține cheia secretă se vor citi de la tastatură sau dintr-un fișier text);
- decripteză o imagine color BMP criptată și salvează imaginea decriptată în memoria externă (căile ambelor imaginii și a fișierului text care conține cheia secretă se vor citi de la tastatură sau dintr-un fișier text);
- afișează pe ecran valorile testului χ^2 pentru imaginea inițială și imaginea criptată, pe fiecare canal de culoare.
- rulează operația de template matching pentru o imagine color BMP și o colecție de şabloane color BMP. Folosiți întotdeauna un prag $p_s = 0.5$ indiferent de şablonul ales. Reuniți toate detectiile rezultate pentru fiecare şablon în parte într-un singur tablou unidimensional D. Căile imaginii și a şablonelor BMP se vor citi de la tastatură sau dintr-un fișier text.
- rulați funcția de eliminare a non-maximelor pe tabloul D și desenați în imagine detectiile rămase folosind o culoare specifică pentru fiecare şablon.