Temă Proiect la Programare Procedurală

Tema 1.

Modulul de criptare/decriptare.

Functiile pe care le-am folosit in acest modul sunt:

- **bmp_lin** -> functia deschide imaginea originala si preia informatii esentiale precum lungimea/latimea in pixeli a imaginii sau header-ul si liniarizeaza imaginea in 3 vectori de culoare (red, green, blue)
- **criptare** -> preia din fisierul text *secret_key* valorea a doua chei secrete : SV si R[0] si se genereaza permutarea folosind algoritmul lui Durstenfeld.(in aceasta functie, pentru a se determina valorile pseudo-aleatoare din vectorul R, se apeleaza functia **xorshift32**.
- **xorare** -> se xoreaza elementele din vectorii redsub, greensub, bluesub conform relatiei de substitutie prezentate la subpunctul d) din prezentarea proiectului.
- **decriptare** -> functia foloseste toti pasii inversi ai functiei **xorare** si **criptare** pentru a reveni la imaginea initiala.
- **chi_test** -> calculeaza distributia valorilor pixelilor pe cele 3 canale de culoare.

Modul de lucru:

Se ia imaginea originala, se prelucreaza cu functiile de criptare si se afiseaza in alta imagine in format bmp. Se iau aceiasi vectori ce contin pixelii criptati si se apeleaza functia de decriptare si se afiseara in memoria externa in alt fisier binar.

Tema 2.

Modulul de recunoașterea de pattern-uri într-o imagine

Functiile pe care le-am folosit in acest modul sunt:

- **grayscale_image** -> transforma imaginea originala in imagine grayscale
- **liniarizare** -> se ia poza grayscale, se afla inaltimea/latimea imaginii din header si se introduc valorile pixelilor intr-o matrice de marime latime*inaltime. La fiecare pixel, se citeste un octet si se sar 2 deoarece au aceeasi valoare (fiind grayscale).
- **corellation** -> calculeaza toate corelatiile unui sablon cu imaginea originala. Pentru fiecare verificare a sablonului, se construieste cate o fereastra si se actualizeaza datele corelatiei. Daca corelatia atinge un anumit prag, aceasta este introdusa intr-o matrice ce are pe liniile si coloanele ei valorile corelatiei.(daca nu se atinge pragul pentru o fereastra, valoarea din matricea de corelatie va fi 0)
- **colorare** -> aceasta functie cauta toate valorile >0 din matricea de corelatie si coloreaza al 3-lea octet din matricea finala (care va fi afisata) de lungime inaltime*latime*3**.**
- **qsort** -> sortam vectorul de detectii

- **nonmax** -> se ia fiecare matrice de corelatie si se cauta fiecare valoare din vectorul de detectii. O data ce este gasita matricea de corelatie din care provine valoarea, aceasta se compara cu toate celealte corelatii gasite in jurul ei (deasupra sau in partea stanga al acesteia) din toate cele 10 matrici de corelatie. Daca valoarea este mai mare**,** valorile mai mici din matricea de corelatie vecine ei devin 0 (respectiv valoarea din vectorul sortat devine 0).
- **afisare** -> fiecare pixel nou din matricea ce urmeaza a fi scrisa (imgcol) primeste anterior o culoare „cod" ce indica culoarea cu care cei 3 octeti trebuie sa fie colorati.

Modul de lucru:

Se transforma imaginea originala in grayscale. Aceasta se liniarizeaza, iar valorile acesteia se copiaza in matricea imgcol. Se transforma apoi toate cele 10 sabloane in sabloane grayscale si se liniarizeaza fiecare in matrici denumite pixsab0, pixsab1,... etc. Se face corelatia pentru cele 10 sabloane si se coloreaza cu o culoare „cod" (denumita CR). Se afiseaza imaginea intermediara (in imaginea exitscale.bmp) care va cuprinde toate ramele cu corelatia peste pragul ps. In vectorul sort_max se iau toate valorile mai mari decat 0.5(ps) din matricile de corelatie si se sorteaza folosind functia quick-sort. Se apeleaza functia nonmax iar apoi se foloseste functia colorare pentru cele 10 matrici de corelatie modficate. Afisam in imaginea grayscale2.bmp imaginea si se poate constata scaderea dramatica a numarului de ferestre.