Extragere imagini bazata pe continut

Student: Izabela Denisa Halas, 30234

anul 2022

# Introducere

* Descrierea problemei abordate
* Contextul problemei / Motivare / Utilitate
  1. **Descrierea Problemei abordate**

Problema extragerii imaginilor avand la baza continutul acestora va fi abordata in cadrul acestei lucrari. Pentru a face acest lucru posibil, este nevoie de o aplicatie cu implementare complexa. Prima specificatie in ceea ce priveste aplicatia este aceea ca, la momentul initializarii, aceasta creeaza o baza de date continand imagini color si care sunt reprezentate prin vectori de nuante predominante. Setul de imagini va fi citit dintr-un director selectat din interfata grafica, odata cu pornirea aplicatiei. Tot in momentul porniri se cere introducerea de catre utilizator a dimensiunii grid-ului M, valoare care trebuie sa fie, fara exceptie, naturala si pozitiva. Ulterior aplicatia permite utilizatorului sa aleaga o imagine color, imagine de intrare pentru care aplicatia va cauta in baza de date cele mai asemanatoare 3 imagini.

In ceea ce priveste efectul aplicatiei, daca a fost lansata cu succes, aceasta va afisa imaginea de intrare, cele 3 imagini din baza de date in ordine descrescatoare a coeficientului de corelare Pearson (avand etichetele: „Similitudine1”, Similitudine”, respectiv Similitudine3”) si histogramele pe canalul Hue asociat imaginilor (imaginea originala si cele 3 similare) obtinute in urma aplicarii filtrului Gaussian.

Mai departe, aplicatia va genera rezultate si in consola, mai exact se vor afisa pentru toate cele 3 imagini atat gradul de similitudine cu imaginea de intrare dupa prima corelare, cat si folosind corelarea Pearson.

In final, la apasarea unei taste pe oricare dintre imagini se va reveni la selectarea unei noi imagini si se va repeta testul.

* 1. **Contextul problemei/ Motivare/ Utilitate**

Aceasta aplicatie poate avea multiple posibilitati de utilizare, fiind foarte versatila si usor de folosit. Daca exista o baza de date de dimensiuni foarte mari continand imagini, aceasta aplicatie va fi capabila sa le gaseasa pe cele care se afla in top 3 cele mai relevante bazandu-se pe calcule complexe efectuate pe pixelii imaginii de referinta.

# Considerații teoretice

* 1. **Metode de rezolvare**

O metoda relevanta pentru solutionarea problemei va fi prezentata in cele ce urmeaza si presupune urmarea unui set de pasi necesari pentru generarea unei rezolvari optime.

Primul pas presupune impartirea imaginii intr-un grid de dimensiune M\*M

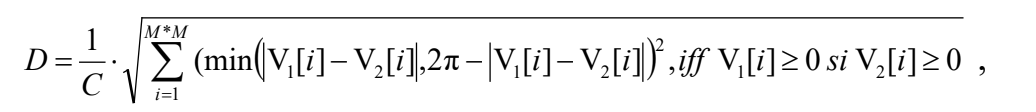
Chart, line chart

Description automatically generated with medium confidence

Apoi pe fiecare zona in parte a grid-ului se face o conversie din RGB in HSI si se calculeaza valoarea Hue medie (μHueЄ[0, 2π)) pe zona, fiind ignorate punctele unde S = 0, si se va stoca intr-un vector V de dimensiune M\*M. In cazul exceptional in care toate punctele dintr-o zona a grid-ului au S = 0, atunci histograma nu are elemente, valoarea medie nu are semnificatie si elementul corespunzator din vectorul V va lua valoarea -1, indicand astfel faptul ca acea portiune trebuie ignorata in viitoarele calcule.

Vectorul V rezultat va fi de fapt vectorul de reprezentare a imaginii.

Daca obiectivul urmarit este acela de a gasi similitudini intre aceasta imagine si o alta imagine, una noua, va fi necesara calcularea vectorului V si pentru aceasta din urma, urmand ca gradul de diferentiere dintre cele doua imagini sa fie dat de formula:



Unde:

, – vectorii corespunzători celor 2 imagini

C = Nπ – constantă de normalizare încât D să aibă valori în domeniul [0,1]

N = numărul de perechi i pentru care se îndeplinește condiția ≥ și ≥ 0

Gradul de similitudine este dat de urmatoarea formula:

S=1-D, dacă N>0 sau S=0, dacă N=0.

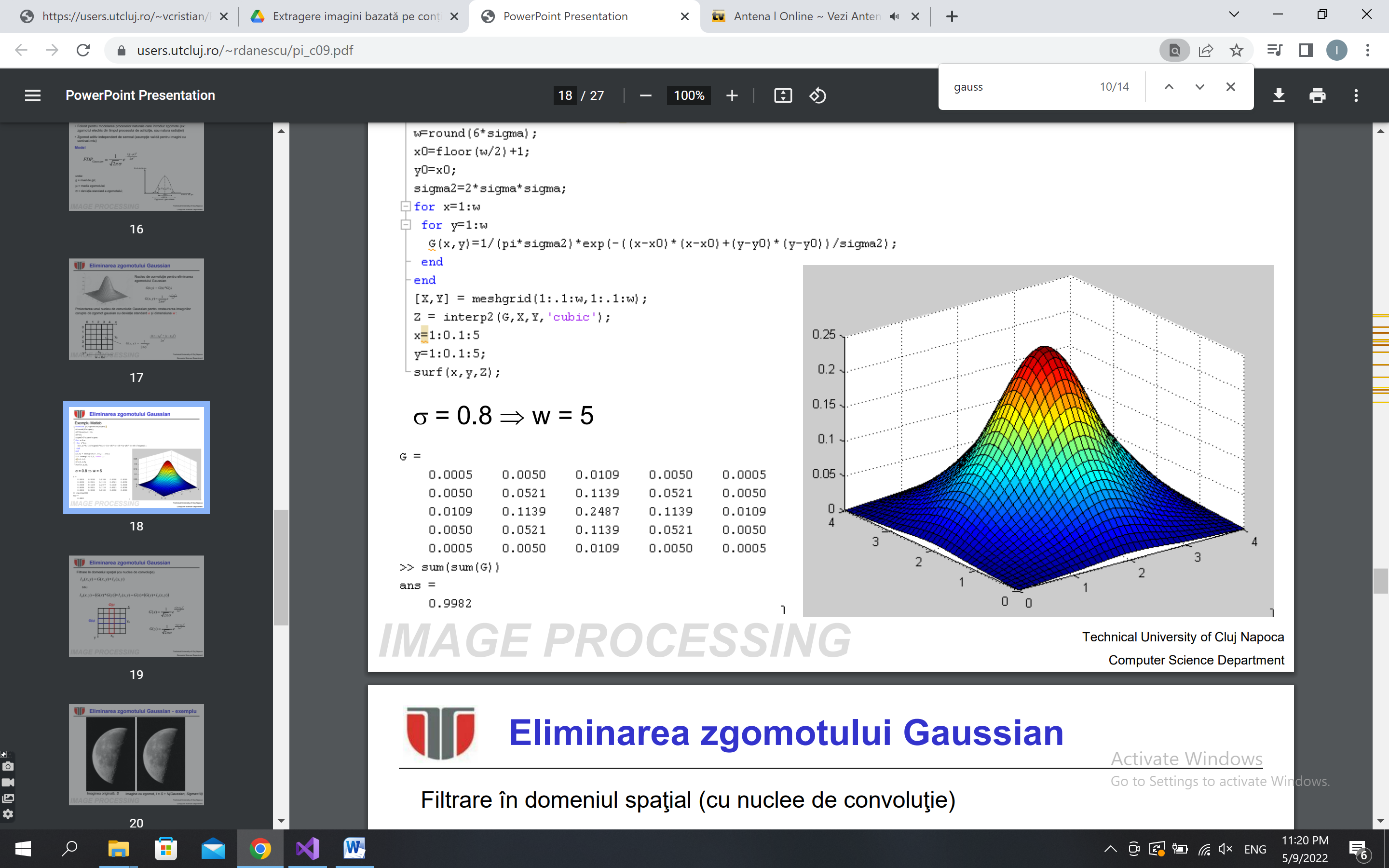
Dacă este cazul unei reprezentări procentuale, S se va înmulți cu 100.

Se poate alege un prag (procentual) de similitudine T :

Graphical user interface, text

Description automatically generated with medium confidence

Filtrul Gaussian pe care l-am ales este acela prezentat in cursul 9. Aplicarea sa se face prin convolutie, asadar „rama” pe care imaginea o va avea va fi influentata de dimensiunea filtrului. Dimensiunea acestuia este de 5x5 si arata asa cum urmeaza:



* 1. **Descrierea teoretica a implementarii**

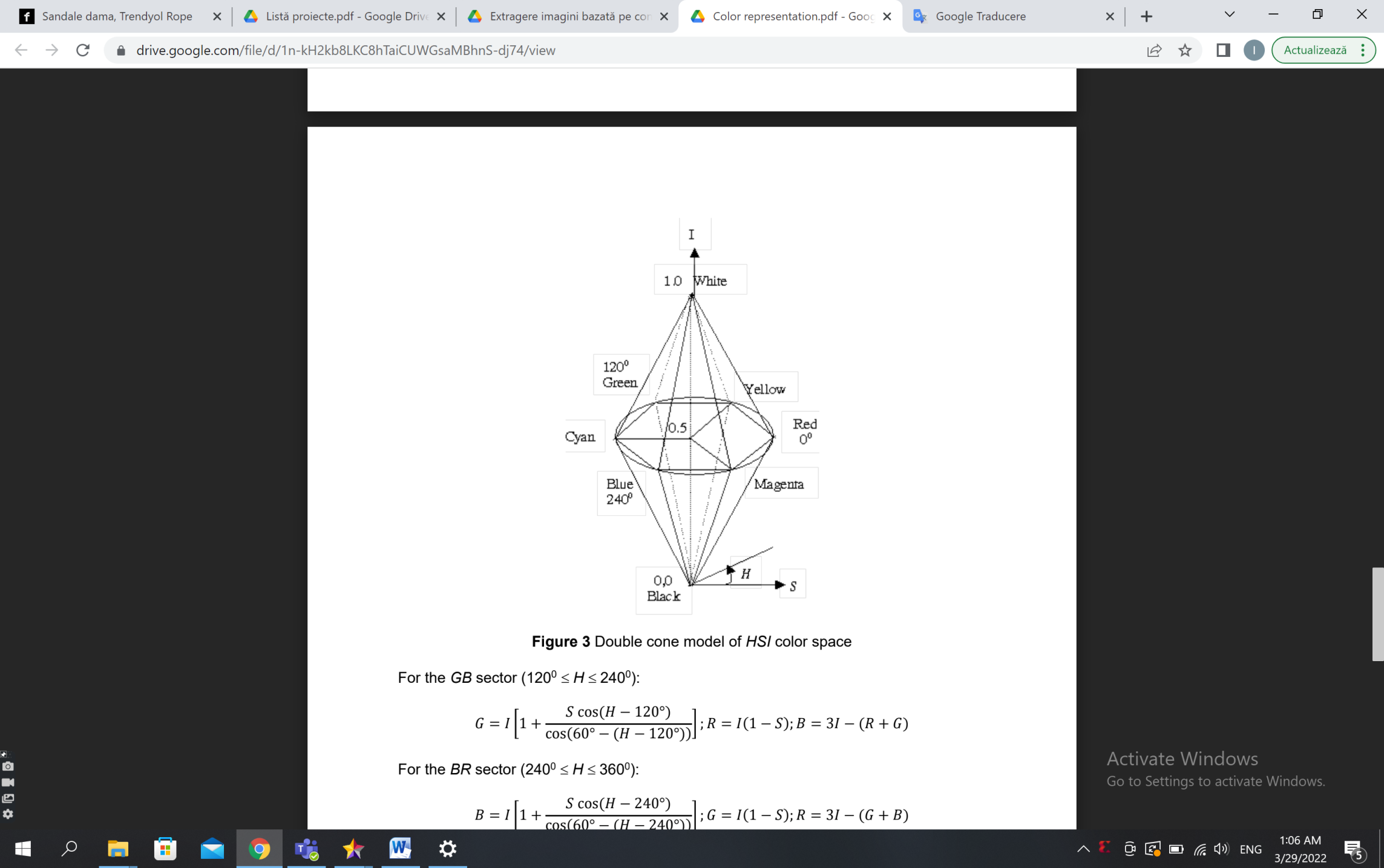
Initial, in elaborarea aplicatiei, se va realiza partea aceea a proiectului care are drept obiectiv generarea unei interfete grafice functionale. Odata ce acest pas este bifat, se va continua implementarea cu algoritmii necesari obtinerii unei functionalitati optime.

Pentru a selecta cele 3 imagini considerate cele mai asemanatoare din intreaga baza de date cu imaginea sursa se va folosi algoritmul prezentat anterior in capitolul

2.1 Metode de rezolvare.

Ulterior se va realiza o rafinare a rezultatului si se vor calcula histogramele pe canalul Hue al spatiului HSI. Pentru a putea realiza acest lucru, este necesara cunoasterea notiunii de spatiu HSI. Principiile HSI spun ca nu este necesara cunoasterea procentelor de albastru si verde pentru a crea o culoare, ci este suficienta ajustarea componentei Hue. Pentru a schimba o nuanta de verde aprins in verde pal se ajusteaza Saturatia, iar pentru a obtine o nuanta mai inchisa sau mai deschisa a unei culori se ajusteaza Intensitatea.

HSI este modelat in coordonate cilindrice, asa ca in figura de mai jos:



Hue (H) este reprezentată ca unghi, variind de la 0 la 360 de grade.

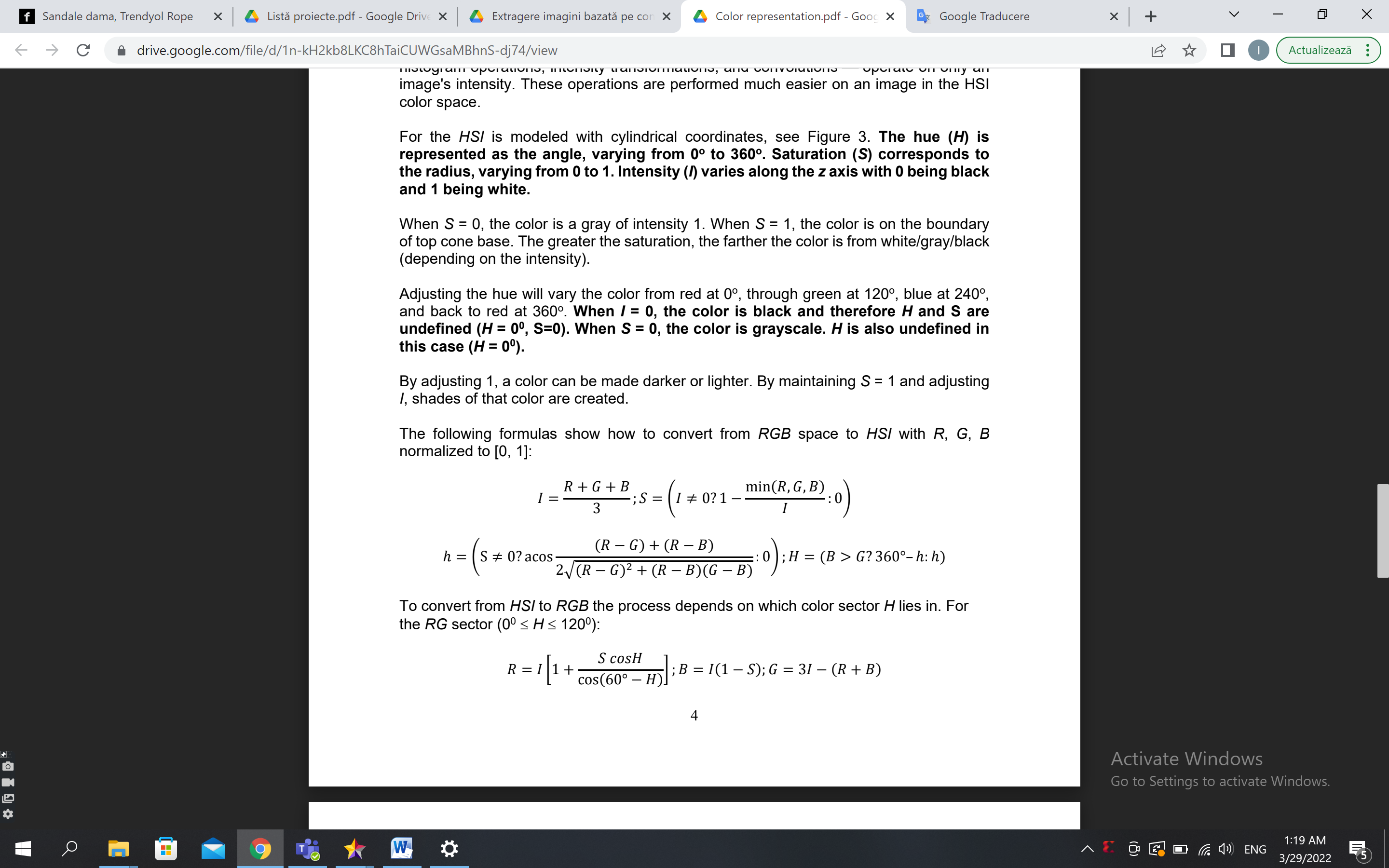
Saturatia (S) corespunde razei, variind de la 0 la 1.

Intensitatea (I) variaza de-a lungul axei z, 0 fiind negru și 1 fiind alb.

Cand saturatia este 0, culoarea este o nuanta de gri de intensitate 1, iar cand saturatia este 1, culoarea se află la limita bazei conului superior. Cu cat este mai mare saturatia, cu atat culoarea este mai îndepartata de alb, negru sau gri, depinzand de intensitate.

Ajustarea Hue va varia culoarea de la rosu (de la 0la 120 grade), la verde (de la 120 la 240 de grade), la albastru (de la 240 la 360 de grade), apoi inapoi la rosu. Cand intensitatea este 0, culoarea este neagră si, prin urmare, Hue si saturatia sunt nedefinite. Când saturatia este 0, culoarea este in nuante de gri, iar Hue este, de asemenea, nedefinit în acest caz.

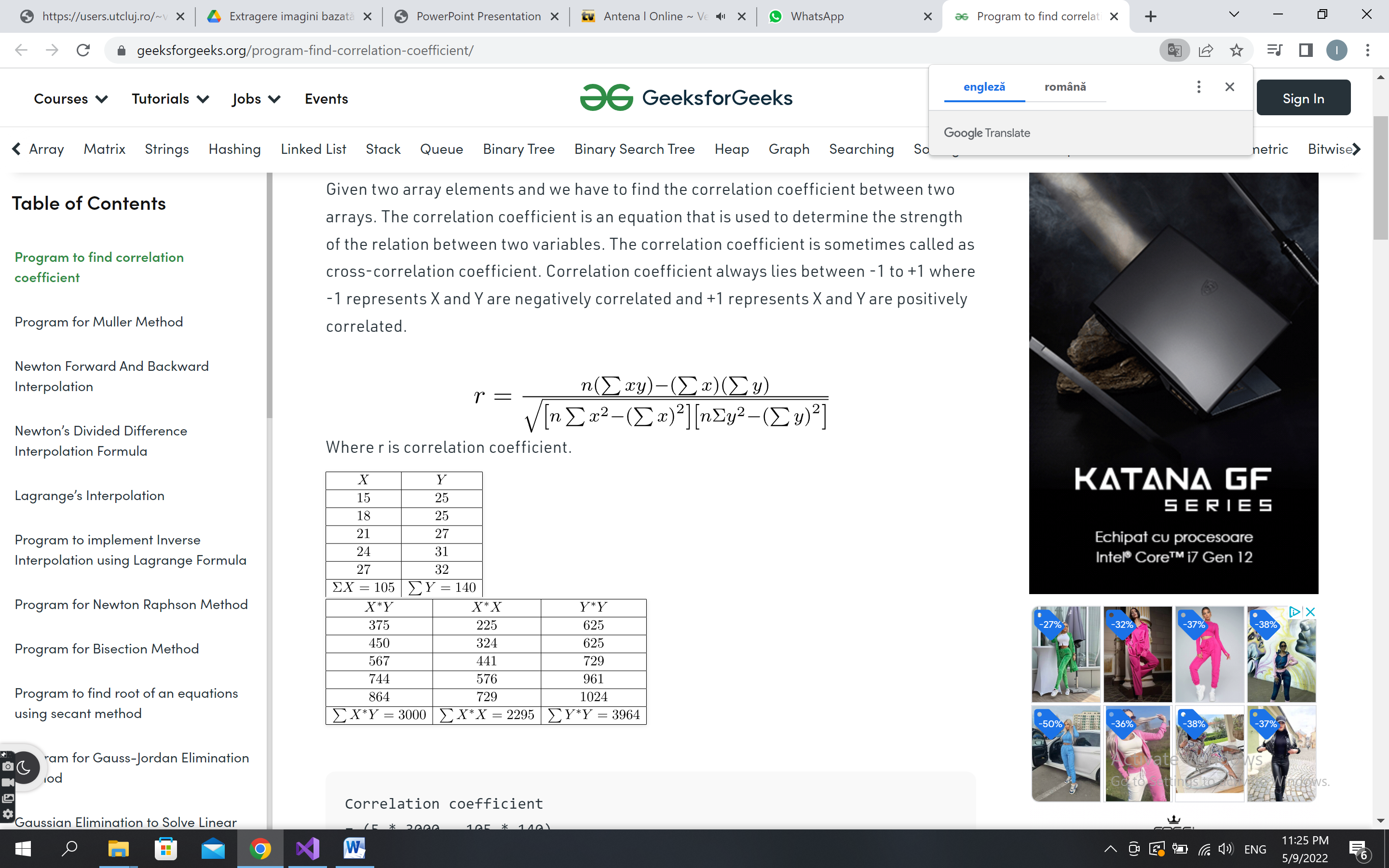
Pentru conversia din RGB in HSI este necesara normalizarea componentelor RGB in intervalul [0,1], apoi trebuie aplicate formulele:



Histogramele mentionate anterior se vor realiza doar dupa ce a fost aplicat un filtru Gaussian de dimensiune 5x5, separat, asupra canalelor RGB ale acestora.

In cele din urma se va determina coeficientul de corelare Pearson al histogramelor rezultate, care este cu atat mai ridicat cu cat imaginile sunt mai similare. Acest coeficient de corelare Pearson, dupa cum ii spune si numele, este o masura a corelatiei liniare intre doua seturi de date, egal cu raportul dintre convarianta a doua variabile si produsul abaterilor standard ale acestora. Se poate observa ca este, de fapt, o masuratoare normalizata a convariantei, astfel incat rezultatul sa fie intotdeauna in intervalul [-1, 1].

Corelarea Pearson este de asemenea importanta in cadrul proiectului, deoarece odata cu aplicarea metodei de similitudine se gasesc cele mai asemanatoare 3 imagini, dar nu se cunoaste, insa, si ordinea acestora. Formula folosita este una simpla, care a fost aplicata pe histogramele imaginilor trecute prin filtrul gaussian, mai exact histogramele canalelor Hue ale imaginilor:



Unde r este tocmai coeficientul Pearson.

# 3. Specificații de implementare

Pana la momentul actual s-a implementat prima parte a proiectului, anume aceea in care se aleg cele mai asemanatoare 3 imagini si se afiseaza pe ecran.

Lansarea aplicatiei deschide consola care asteapta ca intrare intregul 100 pentru a rula proiectul. Odata lansat, proiectul cere selectarea unei imagini pe baza careia se vor face calculele de similitudine. Odata selectata aceasta imagine, apare iar in prim plan consola care cere de data aceasta introducerea unul alt intreg, pozitiv (indicat sa fie 5), acesta reprezentand dimensiunea grid-ului care va fi aplicat peste imagine. Ulterior se va deschide un mic meniu care cere selectarea folderului care se doreste a fi folosit ca baza de date, urmand sa se apese butonul „OK”. Odata ce acesti pasi au fost completati cu succes, in consola trebuie sa apara gradul de similitudine al tuturor imaginilor din baza de date relativ la imaginea aleasa, dar si numarul si inca odata gradul de similitudine al celor 3 imagini obtinute ca fiind cele mai asemanatoare cu imaginea de referinta. Pe ecran vor aparea imaginea aleasa si cele 3 imagini cele mai asemanatoare acesteia.

Pentru inceput am folosit functia de baza din cadrul laboratorului, testOpenImagesFld(), pe care am modificat-o in asa fel incat fiecare imagine citita sa fie si stocata intr-un vector de imagini. Imaginile sunt stocate in ordinea in care au fost citite din fisierul selectat la lansarea aplicatiei. Pentru fiecare dintre aceste imagini, dar si pentru imaginea de referinta, se aplica o functie care separa imaginea in M\*M dreptunghiuri egale intre ele („gridImagine()”).

Functia de separare a unei imagini in M\*M imagini mai mici si egale intre ele presupune parcurgerea imaginii parinte si crearea de bucati ca noi imagini de tipul Mat. Toate acestea vor fi stocate intr-un vector „bucati” de lungime M\*M.

Pe fiecare dintre aceste portiuni mentionate mai sus, ale unei imagini, se aplica o conversie RGB->HSI, insa este relevant doar canalul de Hue, calculandu-se o medie pe baza acestuia. Functia returneaza media. Se va tine un vector pentru fiecare imagine de lungime egala cu numarul de medii relevante obtinute dupa segmentare (de obicei M\*M, exceptie cand o zona din imagine are S=0).

Vectorului de medii Hue al imaginii originale si vectorul de medii al fiecarei imagini din fisierul ales vor fi parametri pentru functia responsabila de calculul similitudinii dintre cele doua imagini „similitudini”. Aceasta returneaza gradul de similitudine dintre 2 imagini. Folosit in conteztul mai multor imagini citite pe rand din fisier, se va crea inca un vector, de data acesta unul care va tine gradele de asemanare dintre imaginea originala si imaginile din folder. Acest vector va fi sortat si se vor lua ultimele 3 imagini din acesta, deoarece acestea vor avea gradul de similitudine cel mai ridicat.

In continuare se va aplica un filtru Gaussian pe canalele RGB ale celor 3 imagini selectionate. Apoi se vor calcula histogramele pe canalul Hue ale acelorasi 3 imagini, dar si histograma imaginii de referinta. Pe baza histogramelor rezultate se va calcula coeficientul de corelare Pearson. Cu cat acest coeficient e mai mare, cu atat imaginile sunt mai asemanatoare intre ele. Cu ajutorul acestuia se va stabili si o ierarhie a clonelor. In functie de aceasta ierarhie vor fi afisate numele pozelor pe ecran.

De asemenea se vor afisa pe ecran si histogramele celor 4 imagini (originalul si cele 3 asemanatoare), iar in consola vor aparea indicii de similitudine si apoi coeficientii de corelare pearson, in ordine descrescatoare.

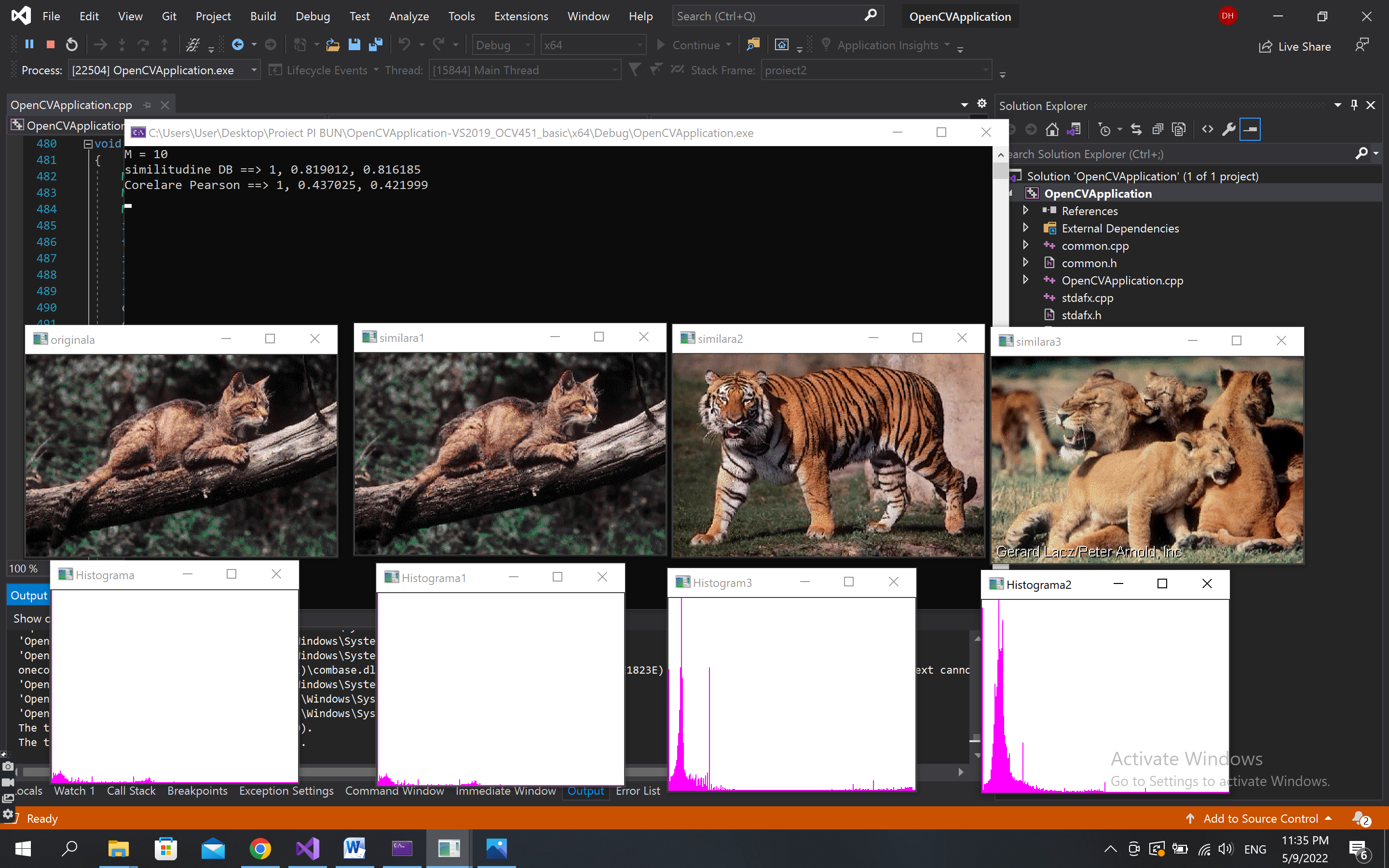
1. Descrierea metodei utilizate (o sugestie ar fi sa realizați o diagramă bloc cu pașii principali ai metodei, urmată de detalierea fiecărui pas in parte)
2. Detalierea conceptelor teoretice / algoritmilor folosiți / secvențe esențiale în pseudocod
3. Ghid de utilizare

# Rezultate experimentale

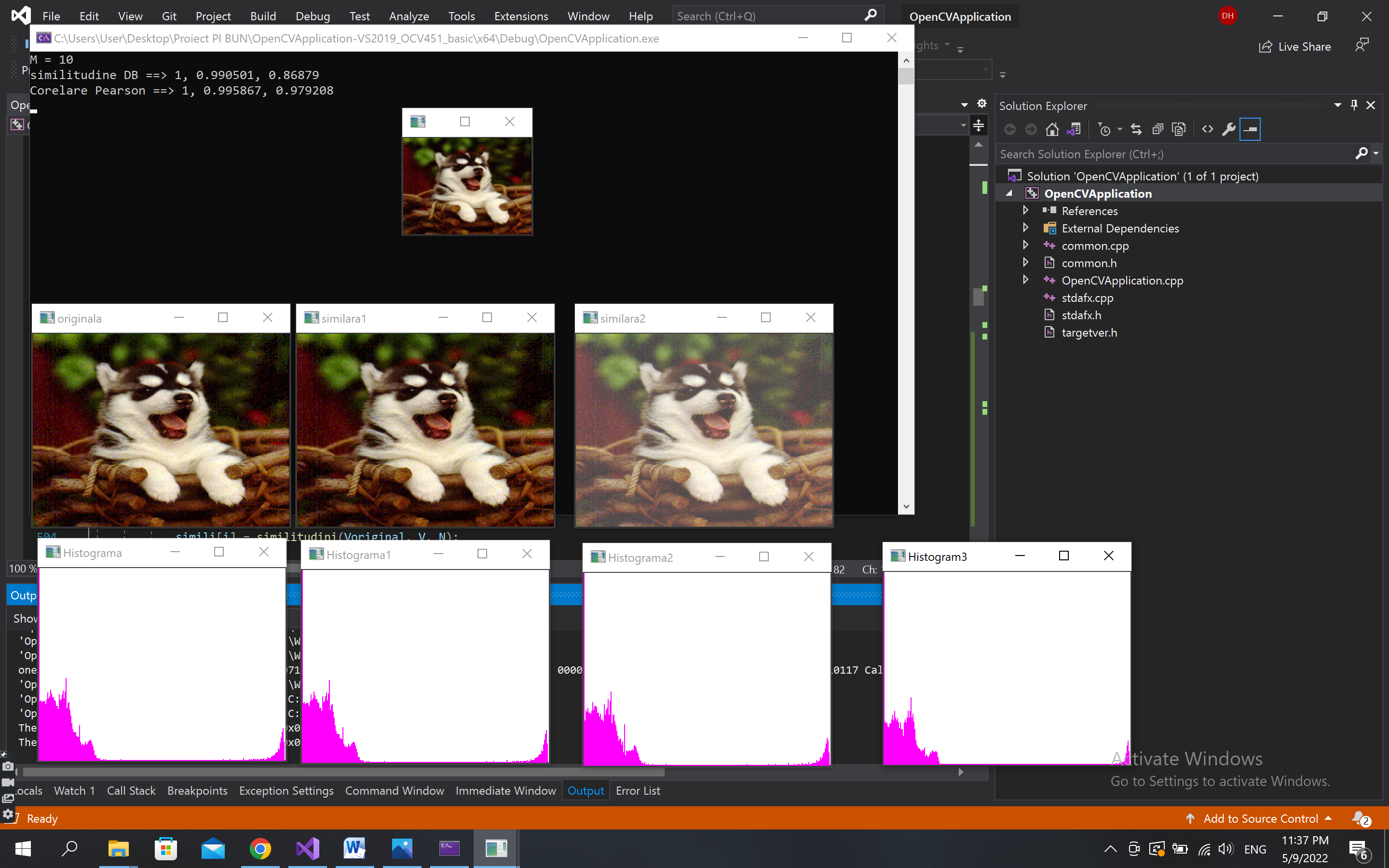
In urma testarii aplicatiei tocmai create, am constatat faptul ca rezultatele sunt asemanatoare cu acelea primite ca si exemplu la inceput. Diferentele care pot aparea sunt datorate catorva zecimale car pot sa difere de exemplele mentionate.

In cele ce urmeaza voi atasa cateva capturi de ecran care sa ilustreze functionalitatea algoritmului:

**Exemplul 1:**



**Exemplul 2:**



**Exemplul 3:**



# Concluzii

In concluzie, proiectul functioneaza in proportie de 90% comparativ cu exemplele primite la inceput, insa chiar si asa rezultatele sunt multumitoare din punctul meu de vedere.

Cerintele au fost atinse in proportie de 100%, cu mentiunea ca unele rezultate pot fi usor diferite de acelea asteptate.

In ceea ce priveste posibilitatile de dezvoltare, acest proiect poate fi modificat in asa fel incat sa functioneze corect in proportie de 100%, ajustand anumite detalii la unele functii (conversia HSI, corelarea Pearson).

# Bibliografie

1. [**https://drive.google.com/file/d/1MgXsGPNo\_1qAuNhPTbxxOkzvwPQ4MPzJ/view**](https://drive.google.com/file/d/1MgXsGPNo_1qAuNhPTbxxOkzvwPQ4MPzJ/view)
2. [**https://drive.google.com/file/d/1d1\_VbqRfohH9AArnJ93tFQyTNYgtk-KJ/view**](https://drive.google.com/file/d/1d1_VbqRfohH9AArnJ93tFQyTNYgtk-KJ/view)
3. [**https://drive.google.com/file/d/1n-kH2kb8LKC8hTaiCUWGsaMBhnS-dj74/view**](https://drive.google.com/file/d/1n-kH2kb8LKC8hTaiCUWGsaMBhnS-dj74/view)
4. [**https://drive.google.com/file/d/1Q6aZ8oOmdlg7JH9AyoEQ7p8Eh29enmrc/view**](https://drive.google.com/file/d/1Q6aZ8oOmdlg7JH9AyoEQ7p8Eh29enmrc/view)
5. [**https://drive.google.com/file/d/1BlXVeB6MQrgsOy6V9ftcu\_B6IVlN1orY/view**](https://drive.google.com/file/d/1BlXVeB6MQrgsOy6V9ftcu_B6IVlN1orY/view)
6. [**https://users.utcluj.ro/~rdanescu/pi\_c09.pdf**](https://users.utcluj.ro/~rdanescu/pi_c09.pdf)
7. [**https://en.wikipedia.org/wiki/Pearson\_correlation\_coefficient**](https://en.wikipedia.org/wiki/Pearson_correlation_coefficient)
8. [**https://www.geeksforgeeks.org/program-find-correlation-coefficient/**](https://www.geeksforgeeks.org/program-find-correlation-coefficient/)
9. [**https://learnopencv.com/cropping-an-image-using-opencv/#cropping-using-opencv**](https://learnopencv.com/cropping-an-image-using-opencv/#cropping-using-opencv)
10. [**https://answers.opencv.org/question/62446/conversion-from-rgb-to-hsi/**](https://answers.opencv.org/question/62446/conversion-from-rgb-to-hsi/)