Definirea problemei

1. Motivare

Detecția de piele în imagini are ca scop identificarea pixelilor sau regiunilor dintr-o imagine care corespund pielii umane. Pe baza acestor regiuni putem îmbunătăți metodele de identificare ale diferitelor părți ale corpului: față, mâini, etc.

Detecția feței e folosită des în practică pentru identificarea automată a persoanelor, ca metoda de autentificare sau în monitorizarea video. Detecția mâinilor și a gesturilor poate facilita interacțiunea om-calculator permițând utilizatorilor să interacționeze cu dispozitive fără sa le atingă. Una din primele aplicații ale detecției regiunilor de piele a fost pentru a identifica imagini pornografice pe internet cu scopul de a le filtra[1]. Alta aplicație în acest domeniu a fost identificare prezentatori în videouri de știri TV pentru adnotare automatică, arhivare și căutare[2].

2. Date de intrare

Pentru acest experiment voi folosi doua baze de date adnotate manual.

Compaq Skin Database[3] conține 13.640 de imagini din care ~4700 conțin piele. Imaginile au fost obținute de pe internet prin intermediul unor programe tip web crawler. Aceste au fost selectate și împărțite în două categorii în funcție de prezența pielii. Pentru imaginile care conțin piele au fost construite măști alb-negru, manual, cu regiunile de piele.

SFA Skin Database[4] a fost construită din combinarea imaginilor din FERET(876 imagini) și AR(242 imagini). Aceasta conține bucăți de piele/non-piele de dimensiuni de la 1 pixel la 35x35. Pentru fiecare dimensiune, SFA are 3354 imagini cu piele și 5590 imagini fără piele.

3. Descriere soluție

Pentru a rezolva această problemă voi combina o metoda de detecție bazată pe culoare și o metoda de detecție bazată pe textură.

Pentru identificarea pixelilor cu o culoare asociată pielii construiesc un dicționar de probabilități (Skin Probability Map) folosind teorema lui Bayes. Astfel pentru fiecare pixel aplic formula $P(S|X) = P(X|S) * P(S) / P(X)$, unde X reprezintă observarea pixelului curent și S observarea pielii. Am ales un spațiu de culoare care nu depinde de modul de iluminare, YcbCr. Am folosit ca date de antrenament imaginile cu piele din Compaq. Autorul a făcut observația că 77% din pixelii din spațiul RGB nu sunt întâlniți deloc în setul de date. Astfel în loc să consider doar un pixel la calculul probabilității am luat maximul din regiune.

Pentru a obține regiuni mai fine am segmentat imaginile înainte de a le aplica clasificare pe culoare. Folosind algoritmul Quickshift am obținut o listă de superpixeli, a căror clasificare am făcut-o după media probabilității pixelilor componenți.

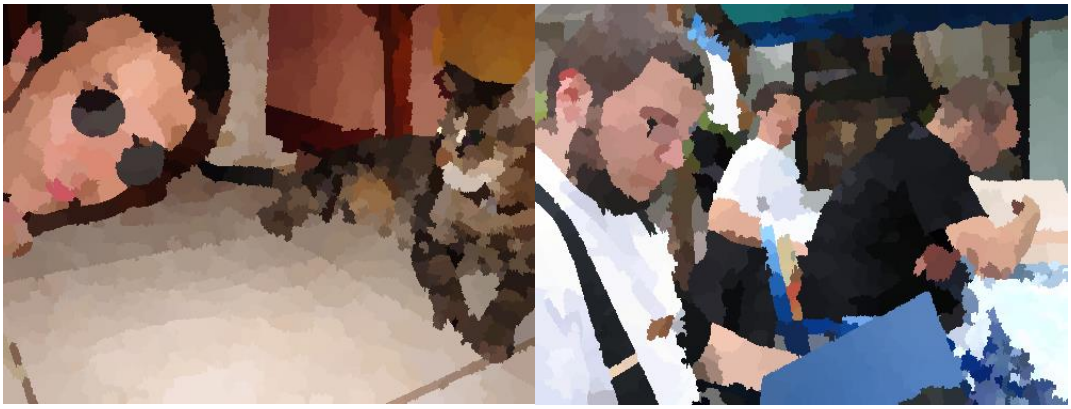
Analiza după textură e o metodă folosită pentru reducere zonelor fals-pozitive. Pentru fiecare imagine din SFA am extras caracteristicile Haralick, folosind librăria mahotas, și am construit un model bazat pe Support Vector Machines, folosind librăria sk-learn. Pentru detecție am parcurs pixelii identificați ca piele la pașii anteriori, am construit o fereastră în jurul lor și i-am clasificat cu modelul preantrenat.

4. Parametraj

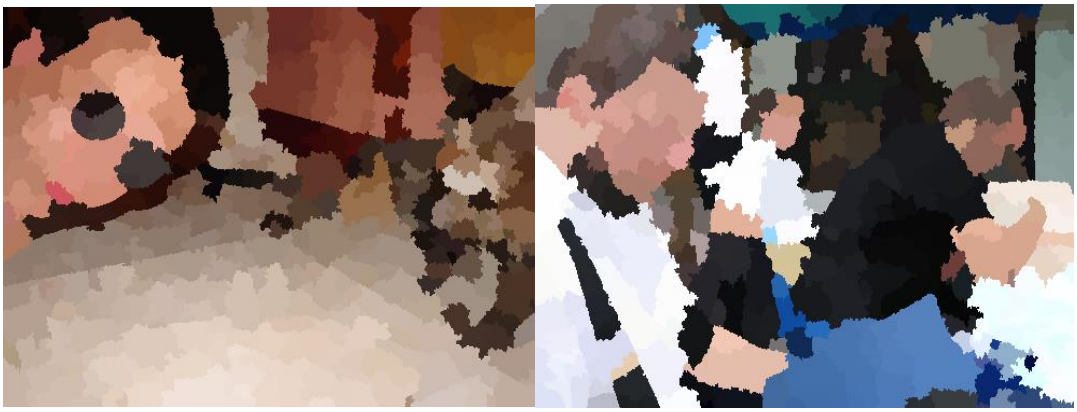
4.1. Segmentare

Pentru segmentarea imaginilor am încercat să identific un set de parametrii pentru algoritmul Quickshift care produc regiuni cât mai mari fără să afecteze conturul obiectelor din imagine. Am făcut următoarele experimente pentru valorile lui sigma și tau:

Sigma = 3, tau = 5



Sigma = 2, tau = 10



Sigma = 4, tau = 8

Am observat că perechea de parametri (3, 5) respectă cel mai bine cerințele.

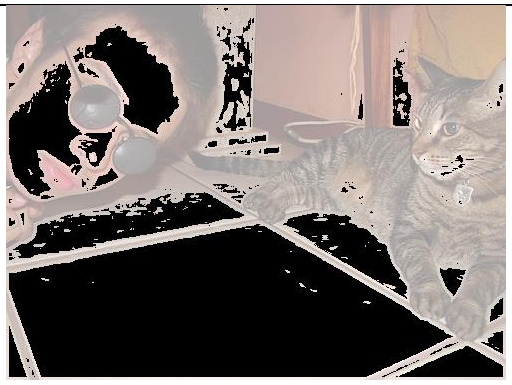
4.2.Detecție culoare

Am testat algoritmul de detecție după culoare pe mai multe spații de culoare: RGB, YcbCr, HSV și am constatat că diferențele de performanță sunt nesemnificative. Această observație a fost făcută și în lucrarea [3], în cadrul algoritmului Bayesian fiind mai important gradul de suprapunere al pixelillor decât spațiul de culoare ales.

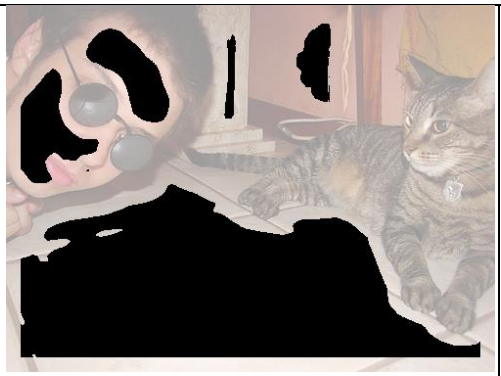
Pentru vecinătatea în spațiul de culoare care să fie analizată pentru fiecare pixel am ales experimental valoarea 4, care acoperă o zonă similară cu cea a histogramelor din [3]. Cele mai bune rezultate pe datele de test le-am obținut cu un threshold între 0.1 și 0.2.

4.3.Detecție textură

Dimensiunea ferestrei am determinato experimental. Zonele negre reprezinta zonele identificate.

7x7		
		

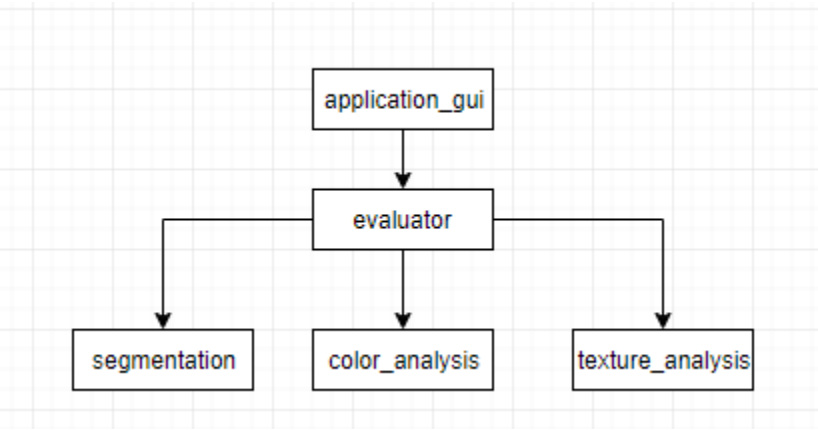
5x5		
		

35x35		
		

Am observat că o fereastră de 5x5 oferă o detecție suficient de bună a zonelor de piele. Pentru ferestre mai mari rata de detecție scade foarte mult.

Documentație de proiectare

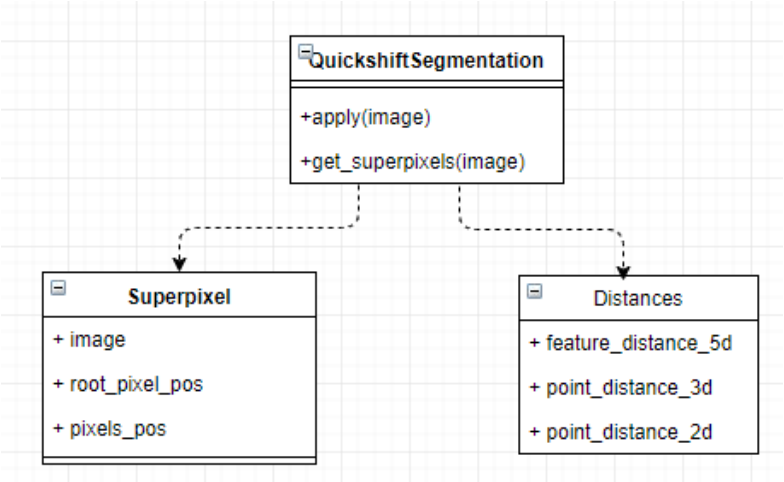
Diagrama modulelor principale:



Deciziile de proiectare care au afectat toate modulele sunt reprezentarea imaginilor folosind tablouri din biblioteca numpy, conform standardelor din biblioteca de prelucrare de imagini opencv și utilizarea ‘namedtuples’ ca și chei în dicționarele folosite.

Toate mesajele privind progresul programului sunt afișate prin intermediul unui logger injectat la pornire. Astfel mesajele pot fi afișate pe consolă, într-un fișier sau redirectate la alt proces.

1. Modulul de segmentare a imaginilor



Funcționalitatea de segmentare a imaginilor este accesibilă prin clasa QuickshiftSegmentation care oferă metodele apply(image), care returnează o imagine în care se pot observa regiunile și get_superpixels(image) care returnează un dicționar cu toți superpixelii identificați și având cheia egală cu rădăcina fiecărui arbore(un arbore conține componentele unui superpixel). Clasa Superpixel încapsulează valorile relevante pentru o regiune, iar clasa Distances oferă câteva funcții pentru calcularea distanțelor între pixeli și între caracteristicile folosite.

2. Modulul de analiză a culorii

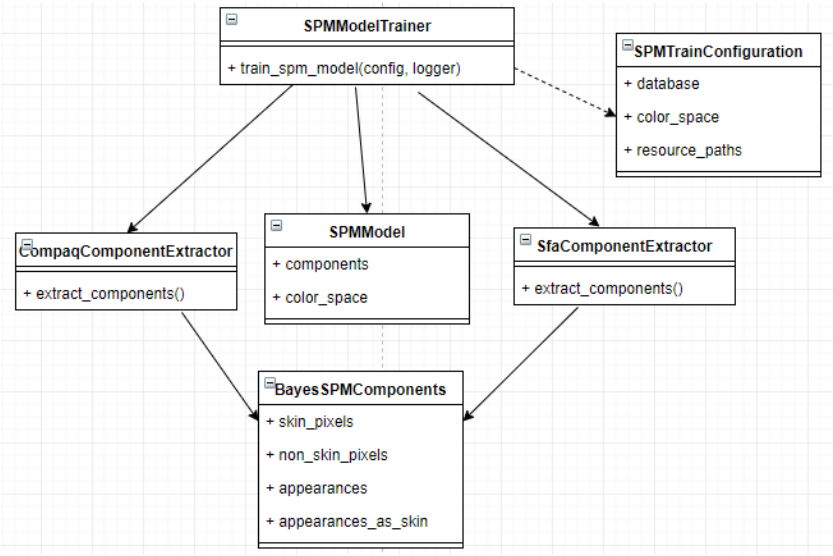
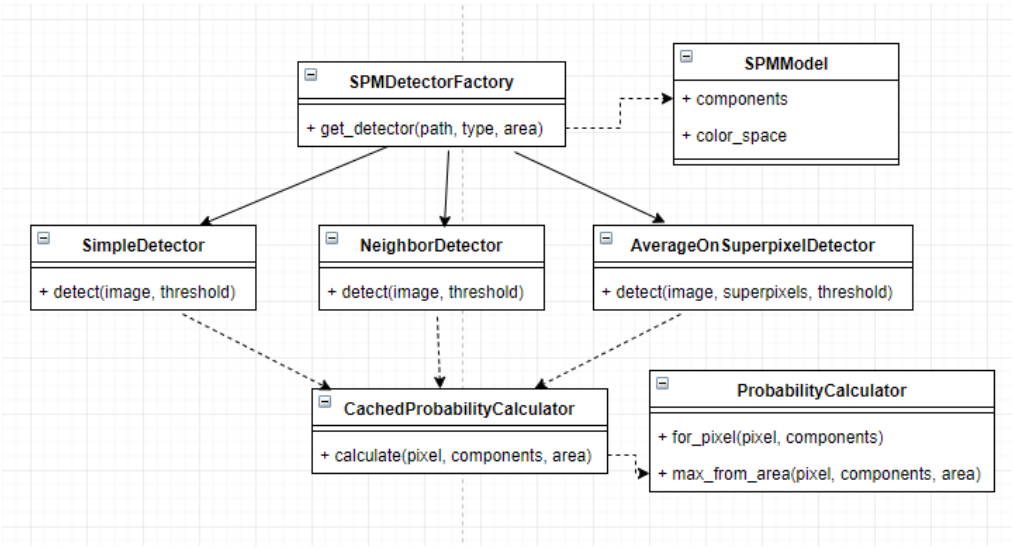
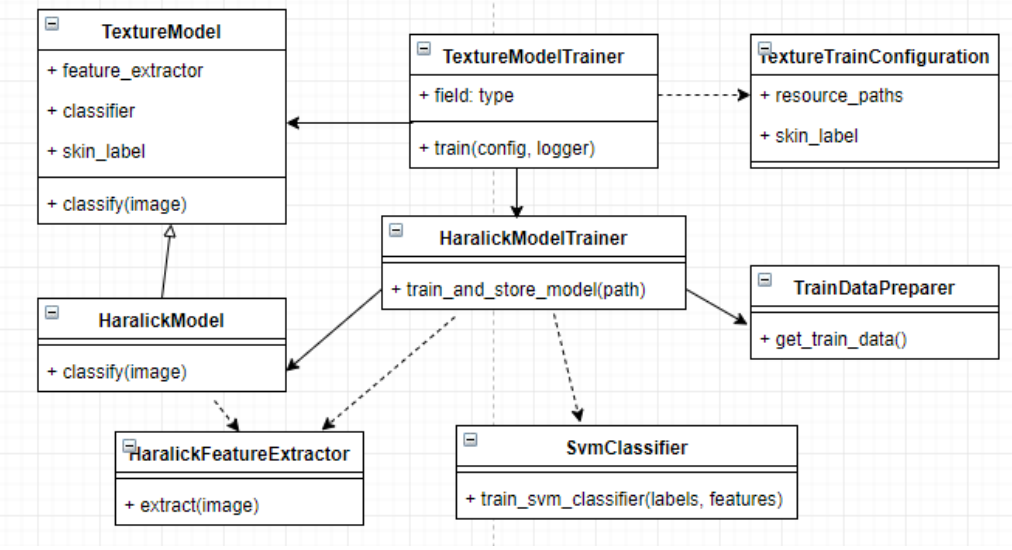


Diagrama anterioară prezintă structura modului de antrenament al modelului. SPMModelTrainer conține o metodă statică de antrenare care primește ca parametru un obiect de tip SPMTrainConfiguration. În funcție de configurația dată se alege un calculator al componentelor relevante pentru metoda Bayesiana: CompaqComponentExtractor sau SfaComponentExtractor. Acestea returnează un obiect de tip BayesSPMComponents care conține: numărul de pixeli de piele, numărul de pixeli non-piele, un dicționar cu numărul de apariții al fiecărui pixel și un dicționar cu numărul de apariții al fiecărui pixel ca piele. Modelul generat de acest modul, SPMModel, reține spațiul de culoare folosit și componentele prezentate anterior.

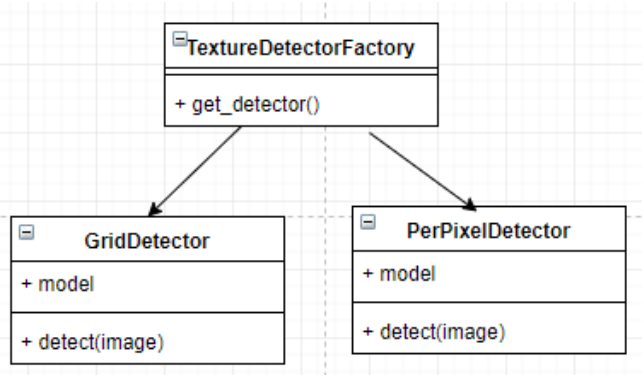


Pentru faza de detecție SPMDetectorFactory alege unul din cei 3 detectori: SimpleDetector(calculeaza probabilitatea pentru un singur pixel la fiecare pas), NeighbourDetector(ia în calcul si probabilitatea pixelilor învecinați) și AverageOnSuperpixelDetector(face o medie a probabilităților pe superpixel). Metodele de calcul a probabilităților sunt definite în clasa ProbabilityCalculator care e accesată prin CachedProbabilityCalculator care oferă caching pentru valorile calculate pentru a crește viteza detectorului.

3. Modulul de analiză a texturii

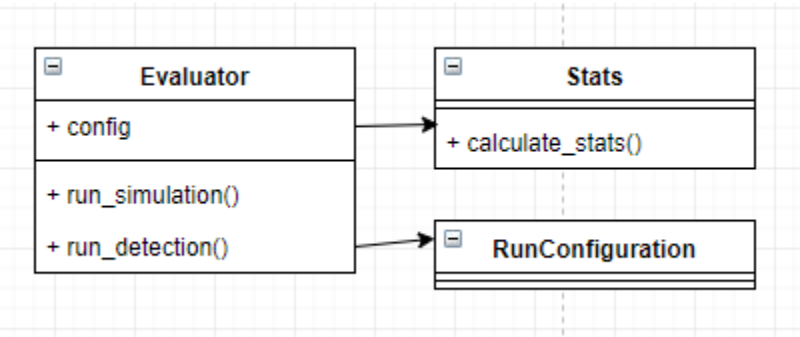


Modelul folosit pentru clasificarea imaginilor după textură, TextureModel, conține un membru care calculează caracteristicile imaginii (feature_extractor), un membru care știe să le interpreteze (classifier) și o etichetă pentru verificarea răspunsului (skin_label). Acesta e specializat de HaralickModel care e antrenat de HaralickModelTrainer și injectat cu HaralickFeatureExtractor, o componentă care extrage caracteristicile Haralick din imagini, și un clasificator de tip SVM antrenat de SvmClassifier, folosind biblioteca sk_learn. TrainDataPreparer încapsulează metoda de extragere a trăsăturilor și rezultatelor așteptate din imaginile de antrenament.



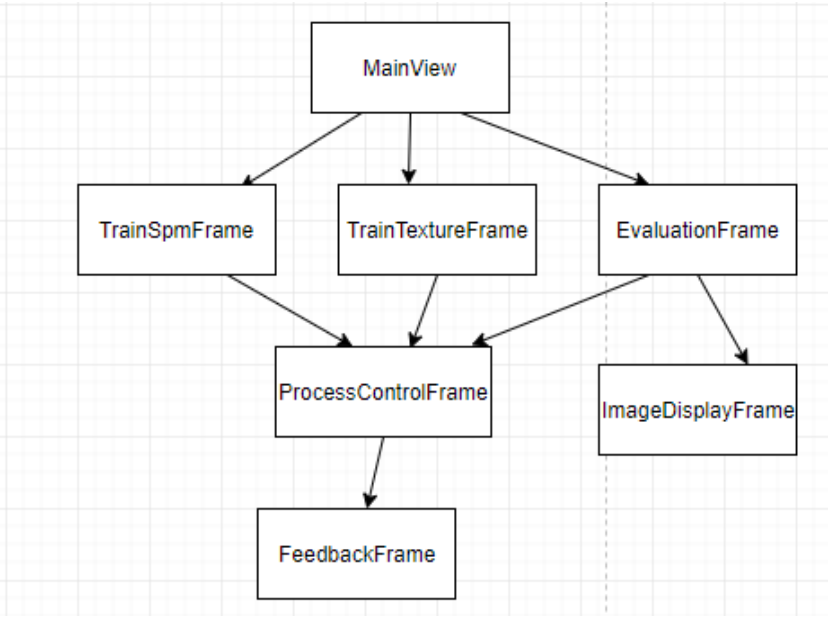
Există două tipuri de clasificatori după textură: GridDetector care împarte imaginea în mai multe blocuri pe care le clasifică independent și PerPixelDetector care construiește o fereastră în jurul fiecărui pixel din imagine.

4. Modulul de evaluare



Evaluator conține două metode: run_simulation(), care rulează algoritmul pe datele de test și folosind clasa Stats evaluează rezultatele, și run_detection() care rulează algoritmul pe imagini noi. RunConfiguration conține toti parametrii configurabili ai modelului.

5. Modulul interfață utilizator



În diagrama anterioară sunt vizibile principalele componente ale aplicației grafice realizată cu biblioteca tkinter. MainView este este o componentă de tip tab-view și conține legături către ferestrele principale: TrainSpmFrame(permite antrenarea modelului bazat pe culoare), TrainTextureFrame(permite antrenarea modelului bazat pe textură), EvaluationFrame(permite evaluarea detectorului de piele). Toate modulele au câte o componentă de tip ProcessControlFrame. Această componentă permite pornirea unui nou proces cu o funcție dată (e.g. antrenarea unui model) și monitorizarea acestuia. Pentru monitorizare, pornește un thread separat care citește mesajele dintr-o coadă și le afișează pe ecran. Mesajele ajung în coadă, din procesul nou creat, prin intermediul unui logger.

Bibliografie

[1] Fleck, M.M., Forsyth, D.A., Bregler, C.: Finding naked people. In: Proceedings of the European Conference on Computer Vision (ECCV). (1996) 593–602

[2] Abdel-Mottaleb, M., Elgammal, A.: Face detection in complex environments from color lmages. In: Proceedings of the International Conference on Image Processing (ICIP). (1999) 622–626

[3] Michael J. Jones and James M. Rehg, "Statistical color models with applications to skin detection," International Journal of Computer Vision, Vol. 46, pp. 81-96, 2002.

[4] CASATI, J. P. B. ; MORAES, D. R. ; RODRIGUES, E. L. L. . SFA: A Human Skin Image Database based on FERET and AR Facial Images. In: IX Workshop de Visão Computacional, 2013, Rio de Janeiro. Anais do VIII Workshop de Visão Computacional, 2013.