UNIVERSITATEA "ALEXANDRU IOAN CUZA" DIN IAȘI

FACULTATEA DE INFORMATICĂ

LUCRARE DE LICENȚĂ

Detectarea oboselii în trafic

propusă de

Ana-Maria Bodnar

Sesiunea: *iulie, 2018*

Coordonator științific

Lect. Dr. Anca Ignat

UNIVERSITATEA "ALEXANDRU IOAN CUZA" DIN IAȘI

FACULTATEA DE INFORMATICĂ

Detectarea oboselii în trafic

Ana-Maria Bodnar

Sesiunea: *iulie, 2018*

Coordonator științific

Lect. Dr. Anca Ignat

Avizat,

Îndrumător Lucrare de Licenţă

Titlul, Numele şi prenumele _____

Data _____Semnătura _____

DECLARAŢIE privind originalitatea conţinutului lucrării de licenţă

Subsemnatul(a) ……

domiciliul în ……..

născut(ă) la data de ……., identificat prin CNP ……., absolvent(a) al(a) Universităţii „Alexandru Ioan Cuza" din Iaşi, Facultatea de …….. specializarea ……., promoţia ……., declar pe propria răspundere, cunoscând consecinţele falsului în declaraţii în sensul art. 326 din Noul Cod Penal şi dispoziţiile Legii Educaţiei Naţionale nr. 1/2011 art.143 al. 4 si 5 referitoare la plagiat, că lucrarea de licenţă cu titlul: _____

_____

_____elaborată sub îndrumarea dl. / d-na _____, pe care urmează să o susţină în faţa comisiei este originală, îmi aparţine şi îmi asum conţinutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licenţă să fie verificată prin orice modalitate legală pentru confirmarea originalităţii, consimţind inclusiv la introducerea conţinutului său într-o bază de date în acest scop.

Am luat la cunoştinţă despre faptul că este interzisă comercializarea de lucrări ştiinţifice in vederea facilitării falsificării de către cumpărător a calităţii de autor al unei lucrări de licenţă, de diploma sau de disertaţie şi în acest sens, declar pe proprie răspundere că lucrarea de faţă nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Dată azi, …… Semnătură student ……

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*Titlul complet al lucrării*", codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza" din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași, *data*

Absolvent *Prenume Nume*

_____

(semnătura în original)

Cuprins

Introducere

Condusul unui autovehicul de către o persoană obosită, care are o stare de somnolență este o activitate periculoasă și ar trebui tratată foarte serios. Majorității oamenilor le este mai degrabă frică și sunt mult mai atenți atunci când este vorba de condusul sub influența alcoolului, nerealizând că și condusul atunci când sunt obosiți este la fel de fatal. La fel ca alcoolul, somnolența încetinește timpul de reacție, scade gradul de conștientizare a mediului înconjurător, provoacă deficiențe în judecată și crește riscul de a fi implicat într-un accident rutier. Însă, cu toate acestea, este aproape imposibil să se determine cu certitudine dacă un accident a fost provocat de oboseala șoferului pentru că nu există teste obiective sau dovezi clare care să ateste cât de obosit a fost șoferul vinovat. Cu toate acestea, există o serie de indicii într-un accident care ajută anchetatorii să-și dea seama că șoferul a adormit la volan. De exemplu, accidentele cauzate de somnolență implică de obicei numai un singur autovehicul în care șoferul este singur. De asemenea, semnele de deplasare sau urmele unor alte manevre, nu există de obicei în astfel de accidente.

Spre deosebire de accidentele în care cauza este prezența alcoolului, în prezent nu există niciun test de sânge, respirație sau orice alt test care să ateste că oboseala este cea care a provocat accidentul.

Conform The National Sleep Foundation[1], printre problemele cauzate de oboseală/ somnolență se numără:

- deficiențe ale timpului de reacție,
- deficiențe de vedere,
- deficiențe în judecată,
- probleme în procesarea informațiilor și în memoria pe termen scurt,
- scăderea performanței, a vigilenței și a motivației,
- scade gradul de conștientizare a mediului înconjurător,
- comportament mai irascibil și mai agresiv

[1] The National Sleep Foundation - http://drowsydriving.org/about/

Conform NHTSA[2], riscul unui accident de mașină sau al unui accident evitat în ultima secundă crește de 4 până la 6 ori dacă șoferul care conduce mașina este obosit.

Un studiu din anul 2017 făcut în 22 de țări a arătat că România ocupă locul 10 în topul accidentelor rutiere produse pe fondul ațipirii la volan, iar potrivit ASCID[3], care citează un studiu al Asociației companiilor franceze de autostrăzi, somnolența este prima cauză a accidentelor mortale pe autostrăzi, în condițiile în care un accident din trei este atribuit oboselii și somnolenței.

2 National Highway Traffic Safety Administration - http://www.nhtsa.gov/DOT/NHTSA/NRD/Multimedia/PDFs/Crash%20Avoidance/Driver%20Distraction/810594.pdf

3 ASCID - Centrul de Informare și Îmbunătățire a Calității Vieții pentru bolnavii de Distrofie Musculară și pacienții asistați Respirator

Contribuții

După cum am precizat în introducere, este foarte greu de "diagnosticat" un accident care a fost provocat de oboseală, iar odată ce acesta a avut loc, sunt destul de greu de rezolvat pagubele produse, atât cele materiale, cât și cele omenești. Mergând pe principiul ”Mai bine să previi decât să tratezi”, această lucrare de licență își propune să încerce să studieze în prima etapă metode pentru prevenirea acestor accidente și mai apoi să încerce o abordare de rezolvare pentru această problemă destul de întâlnită în viața aglomerată, rapidă și obositoare pe care o avem cu toții.

Am structurat teza în 2 părți. În prima parte o să prezint metodele pe care le-am studiat în vederea rezolvării problemei propuse, iar în cea de-a doua parte, o să prezint punerea în practică a două din soluțiile prezentate în prima parte sub o abordare proprie. Metodele pe care o să le descriu se axează în mare parte pe detectarea clipitului și analiza duratei de timp pentru clipiri. Procesul de detectare a clipirilor va fi precedat în primă fază de detectarea feței, iar mai apoi de identificarea ochilor.

Capitolul 1. Metode de a detecta somnolența

Înainte de a prezenta câteva dintre metodele existente pentru studiul oboselii, o să definim noțiunile de oboseală și somnolență. Pe dexonline.com, cuvântul ”oboseală” este definit ca fiind o stare de slăbiciune generală datorată unui efort fizic sau intelectual, pe când ”somnolența” este o stare intermediară între somn și veghe, în care diverse funcții și reacții ale organismului sunt diminuate. Așadar, somnolența nu are neapărat legătură cu oboseala, adică poți avea o stare de somnolență și fără a fi obosit. Deși cele două cuvinte nu sunt întru totul sinonime, pe tot parcursul lucrării le voi folosi ca fiind sinonime pentru că și din starea de oboseală, și din starea de somnolență se poate adormi foarte ușor, și nu ne dorim acest lucru când suntem la volan.

1.1 Mișcarea pleoapelor

Mișcarea pleoapelor este un indicator foarte fiabil pentru detectarea somnolenței șoferului (Erwin, și colab., 1980; Dinges și colab., 1985). Erwin și colaboratorii au studiat diferite metode pentru a determina dacă sunt potrivite pentru problema discutată, inclusiv pletismografia (dispozitiv pentru măsurarea și înregistrarea modificărilor volumului corpului uman sau a unei părți a corpului uman), rata de respirație, electroencefalografia (EEG)[4], caracteristicile electrice ale pielii, electromiografie (EMG)[5], varietatea ritmului cardiac și închiderea pleoapelor. S-a descoperit că închiderea pleoapelor a fost cel mai fiabil indicator pentru debutul somnului dintre metodele enumerate mai sus. Pare destul de evident că dacă pleoapele conducătorului sunt închise, capacitatea de a conduce un autovehicul nu mai există. Skipper și colaboratorii (1984) a studiat capacitatea șoferilor obosiți de a conduce supunându-i la teste de condus de la o oră la o oră jumătate. Au fost intenționat adăugați diverși factori externi pentru a crea o situație cât mai aproape de realitate. S-a constatat că măsurile de performanță, cum ar fi abaterile de la bandă și viteza de deplasare au fost corelate cu închiderea pleoapelor. Pentru înregistrarea comportamentului ochilor în studiile efectuate de Skipper și colaboratori (1984) s-a folosit o camera low light level. În paralel s-a folosit un potențiometru liniar cu care s-a urmărit și înregistrat manual mișcarea pleoapelor participanților.

[4] EEG - înregistrarea în timp a activității electrice cerebrale
[5] EMG – examinare a sistemului nervos periferic și a sistemului muscular

1.2 Mișcarea ochilor

Pentru detectarea mișcărilor ochilor în timpul somnului sau înainte de a dormi există două metode generale. Prima metodă este Rapid Eye Movements (REMs), iar a doua metodă se bazează pe faptul că la majoritatea participanților declanșarea somnului este însoțită de mișcări lente ale ochilor (Carskadon, 1980). Mișcările lente și relativ circulare ale ochilor precedă adormirea. Acest fenomen are loc și la etapa 1 a somnului normal în timpul nopții. Caracteristicile mișcărilor ochiului uman sunt în dependență de nivelul de vigilență a persoanei. S-a dovedit că mișcările lente ale ochilor (SEM) este unul dintre semnele cele mai caracteristice ale fazei de tranziție dintre starea de veghe și cea de somn (Planque si colab., 1991). La un participant complet treaz se pot observa mișcări rapide ale ochilor, pe când la un participant somnoros, ochii au o mișcare de pendul de la stânga la dreapta, iar numărul mișcărilor rapide începe să se diminueze (Hiroshige și Niyata, 1990). Mai multe SEM-uri sunt detectate în stadiul 1 de somn, dar ele apar și în restul perioadei de somn, separând trezirea de somn.

1.3 Activitatea musculară

Electromiograma (EMG) este o înregistrare pentru activitatea electrică a mușchilor activi, în special a mușchilor faciali. Această activitate poate fi, de asemenea, înregistrată de electrozi care sunt puși pe suprafața feței. La om, pentru EMG sunt luați în considerare mușchii de sub bărbie deoarece mușchii din această zonă prezintă schimbări dramatice asociate etapelor de somn (BOSB, 1997). Chiar și atunci când o persoană este total relaxată, activitatea mușchilor din acea zonă poate fi observată. Acest lucru se datorează faptului că fiecare mușchi este compus din multe fibre care sunt conectate de nervi. Când un fibrele dintr-un mușchi sunt activate prin inervații nervoase, o schimbare a potențialului electric poate fi văzut. Atunci când mușchiul este relaxat, mai puțini nervi sunt activați, deci un EMG mai mic este înregistrat. Electromiograma este folosită pentru a prezice somnolența ajutându-se de diferiți mușchi faciali.

1.4 Activitatea creierului

Electroencefalograma (EEG) a fost descoperită în 1929 de Hans Berger, psihiatru elvețian. El a descoperit că există mici schimbări de tensiune între electrozi atunci când au fost puși în contact cu scalpul. Modificări de voltaj sunt amplificate și examinate pentru diferite durate de timp. Baza fiziologică exactă a variațiilor de tensiune nu este în întregime cunoscută, dar se crede că în mare parte, acestea provin în din modificările de tensiune ale membranelor

celulare nervoase. Erwin și colab. (1980) a constatat că nu există nicio modificare fiabilă a activității creierului înainte de închiderea pleoapelor, dar la închiderea acestora, cercetătorii au descoperit că o schimbare foarte rapidă în tiparele undelor cerebrale are loc. Această schimbare este identificată ca etapă timpurie a somnului. Totuși, Planque si colab. (1991) susțin că schimbările clare în activitatea creierului sunt observate pe durata trecerii de la starea de vigilență, la hipoalergenitate, somnolență și mai apoi somn. O încetinire a activității creierului, urmată de o creștere a procentului de unde alfa, apoi de o scădere a procentului de unde beta se observă atunci când vigilența este în scădere.

1.5 Schimbări de dimensiune a pupilei

Mișcările naturale ale pupilei în întuneric au fost descrise că ar reflecta oboseală, somnolență. Modificările în stabilitatea pupilei și gradul de oscilații au fost găsite ca fiind normale la persoanele obosite. Comportamentul pupilelor la oameni sugerează că acțiunile pupilei reflectă evenimente autonome, și în consecință, este un indicator indirect, dar precis pentru somnolență (Sharon si colab. 1996). [8]

Acestea sunt câteva dintre metodele existente pentru detectarea somnolenței. În această lucrare o să mă axez și o să studiez mișcarea pleoapelor, și anume clipirile.

Capitolul 2. Metode pentru a măsura somnolența

Ca și metode pentru a ”măsura” somnolența, două dintre cele mai cunoscute și folosite metode sunt PERCLOS și The Johns Drowsiness Scale (Johns și colab., 2003). Metoda din urmă se bazează pe o combinație de variabile, inclusiv rapoarte de amplitudine, viteză pentru închiderea pleoapelor. Sunt calculate schimbările poziției pleoapelor în timpul unei clipiri. Prima metodă o să fie detaliată în continuare. Ambele metode măsoară clipirile ochilor utilizând diferite variabile care țin fine de dimensiunea clipirilor, varianța acestora, etc.

PERCLOS (PERcentage of CLOSure)

PERCLOS (PERcentage of CLOSure) este o măsură pentru detectarea vigilenței șoferului care a fost identificată ca fiind cea mai fiabilă și valabilă metodă într-un studiu realizat de Administrația pentru Autostrada Federală a Statelor Unite. Diferiți autori consideră acest procent ca un standard pentru detectarea somnolenței.

PERCLOS este procentul de timp în care ochiul este închis (pleoapele acoperă pupila și reflectă închiderea lentă a pleoapelor (Dingers & Grace, 1998). Metrica de oboseală PERCLOS a fost stabilită în 1994 într-un studiu în care se simula condusul unui autovehicul, ca fiind proporția de timp dintr-un minut în care ochii sunt închiși cel puțin 80%, aproximativ 48 de secunde (Wierwille și colab., 1994) [6]. Ochii larg deschiși reprezentă 0%, iar ochii închiși reprezintă 100%. Administrația Federală a autostrăzilor (FHWA) și Administrația Națională pentru Siguranța Traficului pe Autostrăzi din SUA (NHTSA) ia în considerare PERCLOS ca fiind una dintre cele mai promițătoare și vigilente măsuri în timp real pentru detectarea somnolenței în vehicule. Astăzi există trei măsuri PERCLOS în uz:

- P70 – ochii sunt închiși cel puțin 70% într-un minut;
- P80 – ochii sunt închiși cel puțin 80% într-un minut;
- EM (EYEMEAS) - procentul mediu pătrat al frecvenței de închidere a pleoapelor.

$$PERCLOS = \frac{Timpul\ în\ care\ ochii\ sunt\ închiși}{Timpul\ total}$$

Trebuie remarcat faptul că în studiul lui Wierwille și colab. (1994) și studiile tehnice conexe de la Administrația Autostrăzilor Federale, fața participantului a fost monitorizată și înregistrată pentru a detecta închiderea pleoapelor și apoi o persoană a văzut înregistrările și a evaluat gradul în care au fost ochii șoferilor închiși din moment în moment. Provocarea referitoare la valorile PERCLOS este măsurarea automată a poziției pleoapelor; iar încercări de succes privind această măsurare a poziției pleoapelor (iar mai apoi obținerea valorii PERCLOS din aceasta) sunt raportate de Dinges & Grace, (1998), unde o cameră CCD[6] monitorizează fața șoferului. Valorile PERCLOS sunt măsurate direct și estimate prin metode non-parametrice pentru detectarea somnolenței la șoferi. Dinges & Grace (1998) a folosit componenta conectată și SVM[7] pentru a verifica când clipește ochiul [7].

Cu toate că această metodă este cea mai utilizată măsură pentru a stabili dacă ochii monitorizați sunt sau nu obosiți și există riscul de a adormi, aceasta metodă are și unele puncte slabe. Metoda aceasta da rezultate bune atunci când pleoapele au o mișcare lentă, normală, dar pentru clipirile foarte rapide (durată de aproximativ 100 ms) PERCLOS nu dă rezultate prea bune. Clipirile rapide sau micro clipirile sunt o măsură importantă pentru detectarea micro somnului, adică a unui episod temporar de somn care poate dura de la o secundă până la 30 de secunde, timp în care persoana este inconștientă și nu poate răspunde la diferite impulsuri, iar PERCLOS nu poate detecta acest tip de somn (Hargutt, 2003).

[6] CCD – (charge-couple device) Dispozitivul cu cuplaj de sarcină este un registru cu deplasare analogic ce permite transportul semnalelor analogice prin mai multe etape succesive, sub controlul unui cuplaj de ceas
[7] SVM – Support vector machine

Model de rețea Bayesiană despre oboseală descrisă în lucrarea [14]

Capitolul 3 – Metode pentru studierea mișcării pleoapelor

În acest capitol voi aborda câteva dintre metodele ce există la momentul actual pentru detectarea somnolenței la volan, punând accent pe metodele pe care le-am implementat, implementarea urmând să fie prezentată în partea a doua a lucrării. Dintre metodele prezentate în primul capitol, mă voi axa pe prima metodă, mișcarea pleoapelor care implică clipirile. Am ales acestă modalitate deoarece cred că este cea mai comodă și mai practică, atât pentru conducătorul auto, cât și pentru cei care studiază asta. De exemplu ar fi destul de incomod și costisitor dacă în timp ce șoferul este la volan, acesta ar avea tot felul de senzori atașați de față pentru a vedea activitatea musculară, sau cea a creierului.

3.1 Despre clipire

Corneea este unul dintre organele cu cea mai bună vascularizare. Aceasta trebuie să fie în permanență umezită, pentru că, în caz contrar vederea devine încețoșată. Lacrimile produse de glanda lacrimală curăță ochii și îi apără împotriva agenților patogeni.

Frecvența clipirilor normale depinde de sistemul nervos simpatic. Alfred Adler[8] a observat că timpul între clipiri durează 2.8 secunde pentru bărbați și 4 pentru femei. Timpul de clipire durează între 0.3 și 0.4 secunde și depinde de emoții și de ceea ce văd oamenii.

Clipirea este închiderea si deschiderea rapidă a pleoapelor într-un mod parțial subconștient. În acest proces sunt implicați mai mulți mușchi, doi mușchi principali sunt orbicularis oculi și levator palpebrae superioris care controlează închiderea și deschiderea ochilor. Există două tipuri de clipire involuntară (clipitul spontan și clipitul reflex) și clipitul voluntar.

Clipitul spontan este acea acțiune pe care nu o observăm (dacă nu începem să ne gândim la asta) și care ne ajută sa menținem globul ocular curat și umed.

Clipitul reflex este răspunsul la un eveniment neașteptat. De exemplu, clipim reflexiv atunci când un obiect se apropie de ochi sau când auzim zgomote puternice. Acest tip de clipit durează cu o fracțiune de secundă mai mult decât clipitul spontan și este un mecanism de protecție pentru a preveni deteriorarea globului ocular de potențiale amenințări.

8 Alfred Adler (1870 – 1937) – doctor și psiholog austriac, fondator al școlii de psihologie individuală

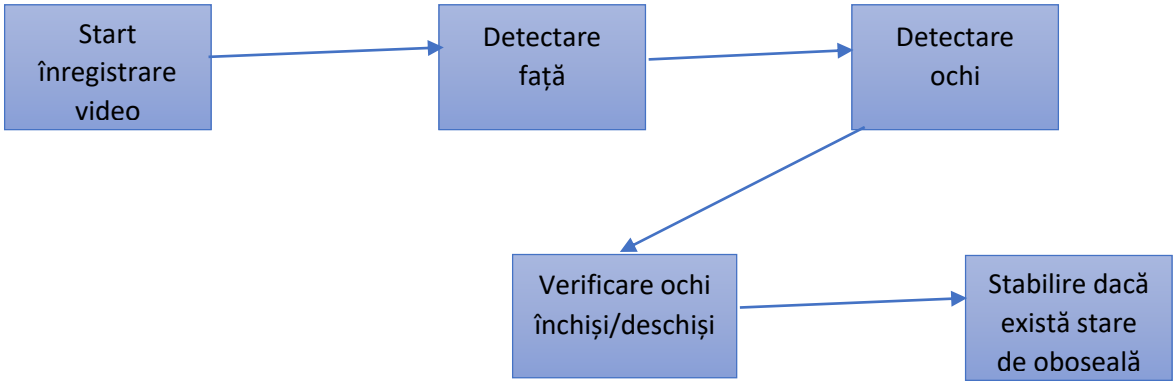
Clipitul voluntar reprezintă acțiunea pe care o facem în mod conștient (deliberat). De asemenea, și acest tip de a clipi durează mai mult decât clipitul spontan. [3]

Dacă avem la dispoziție cunoștințe despre cum să interpretăm clipirile în unele situații, acestea ne pot da mai multe informații decât am crede. Mișcările pleoapelor depind de emoțiile persoanei respective. Potrivit informațiilor despre frecvența clipirilor, putem recunoaște daca cineva este plictisit, obosit, se gândește foarte intens la ceva anume, minte sau are o problemă medicală cu ochii. De exemplu, când cineva minte, clipirile sunt mai lente pentru că creierul lucrează mai mult decât de obicei pentru a crea scenariile neadevărate. După ce minciuna a fost spusă, clipirile sunt mai dese. Când oamenii nu mint, ochii lor au clipiri care au o regularitate anume, normală. [10]

Indiferent dacă ne dorim să vedem dacă o persoană minte, sau este obosită, sau pur și simplu dacă este bolnavă, trebuie în primă fază, pentru toate cazurile să detectăm clipirile. În continuare vom detalia acest procedeu.

3.2 Cum detectăm clipirile?

Pentru a detecta clipirile, trebuie să găsim prima dată ochii, iar pentru a localiza ochii, trebuie să detectăm fața. În diagrama de mai jos voi schematiza acest proces.



3.3 Detectarea feței

În continuare, o să enumăr câteva metode pentru detectarea feței și a ochilor, urmând a descrie mai detaliat metoda folosită de către mine în implementarea ce urmează a fi prezentată.

Metoda	**Autor**
Template matching	(Eriksson and Papanikolopoulos 1998)
Gabor wavelets	(Sirohey and Rosenfeld 2001)
Support Vector Machines	(Liu, Xu and Fujimura 2002)
Projection Functions	(Zhou and Geng 2003)
Principal Component Analysis	(Wang, et al. 2005)
Combined binary edge and intensity information	(Song, Chia and Liub 2006)
Pixel to edge information	(Asteriadis, et al. 2006)
Geometrical information	(Asteriadis, et al. 2006)
Neural Networks	(D'Orazio, et al. 2007)
Classifiers based on Haar-like features	(Hadid, et al. 2007)
Camshift prediction in YCbCr plane	(Hong and Qin 2007)
Skin color segmentation	(Liying and Haoxiang 2008)

Tabel 1. Metode populare de detecție a feței și ochilor[11]

Sunt multe moduri de a detecta fața, iar câteva din cele mai populare se găsesc în tabelul anterior. Nu contează neapărat metoda pe care o folosim pentru a localiza fața, este în funcție de preferințele fiecăruia. Fie că folosim OpenCV cu haar cascades, sau Linear SVM object detector împreună cu Histogram of Oriented Gradients și Object Detection, sau folosim chiar un algoritm de deep learning pentru a localiza fața în secvența video pe care o avem, important este să folosim metode care ne ajută să obținem coordonatele (x, y) ale feței.

Capitolul 4. Algoritmi pentru detectarea clipirilor

În acest capitol voi prezenta câțiva algoritmi pentru detectarea clipirilor. Pentru a aplica aceste metode se consideră că în prealabil, s-au aplicat diverși algoritmi pentru a calcula coordonatele carteziene ale feței, și mai apoi ale ochilor.

4.1 Primul algoritm

Primul algoritm despre care o să vorbesc este unul foarte simplu în esență și care dă randament destul de bun la prima vedere. Este propus de Tereza Soukupova și Jan Cech de la Facultatea de Inginerie Electrică de la Universitatea Tehnică din Praga. Documentul [9] descrie în de aproape acest algoritm.

Algoritmul se folosește de facial landmark detaliat în capitolul 5. După ce se cunosc coordonatele celor 6 puncte care sunt distribuite pe conturul ochiului, se calculează un raport cu ajutorul căruia se pot trage concluzii cu privire la starea ochiului, deschis sau închis.

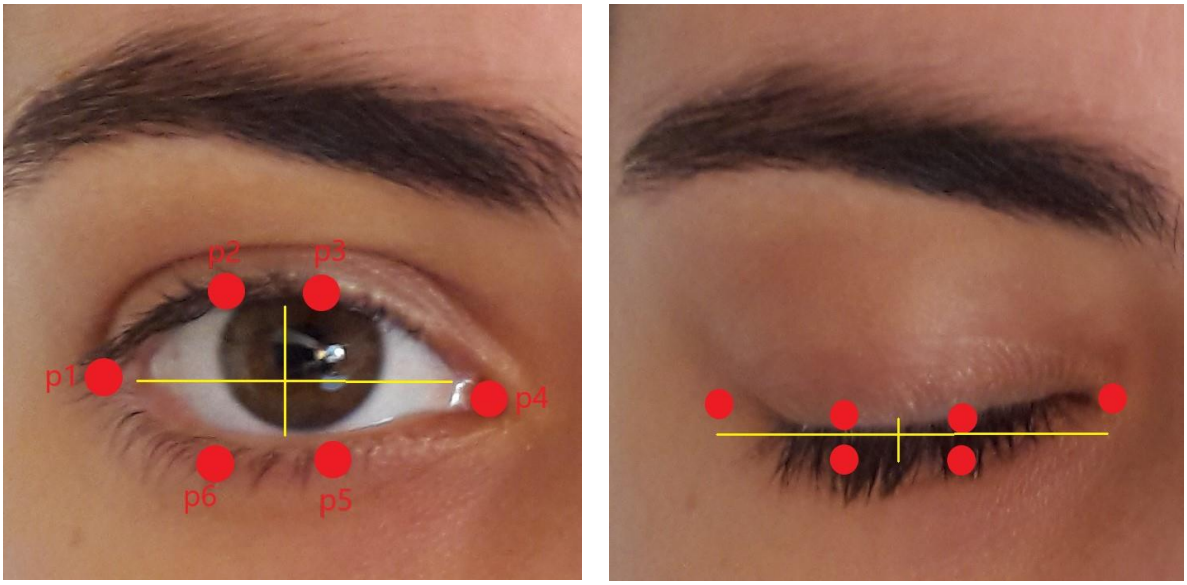


Figura1. Ochi deschis/închis adnotat automat cu punctele p[i]

Pentru fiecare cadru (frame) din secvența video, sunt detectate coordonatele punctelor de pe conturul ochilor. Cu ajutorul acestor puncte, este calculat raportul (EAR[9]) care descrie starea ochiului la acel moment.

[9] EAR – eye aspect ratio

$$EAR = \frac{\| p_2 - p_6 \| + \| p_3 - p_5 \|}{2 \| p_1 - p_4 \|}$$

Formula1 – Raport EAR

Unde p1, p2, p3, p4, p5, p6 sunt coordonatele carteziene ale punctelor din formula1.

Raportul EAR este în mare parte constant atunci când ochii sunt deschiși, iar când ochii sunt închiși, acest raport de apropie de zero. EAR nu depinde de persoană sau de poziția capului. Valoarea calculată pentru un ochi deschis variază destul de puțin de la persoană la persoană și este complet invariant la o scalare uniformă a imaginii și la rotirea în plan a feței. Atunci când persoana clipește, clipește cu ambii ochi în același timp, așa că se va face media aritmetică pentru valorile EAR a celor 2 ochi.

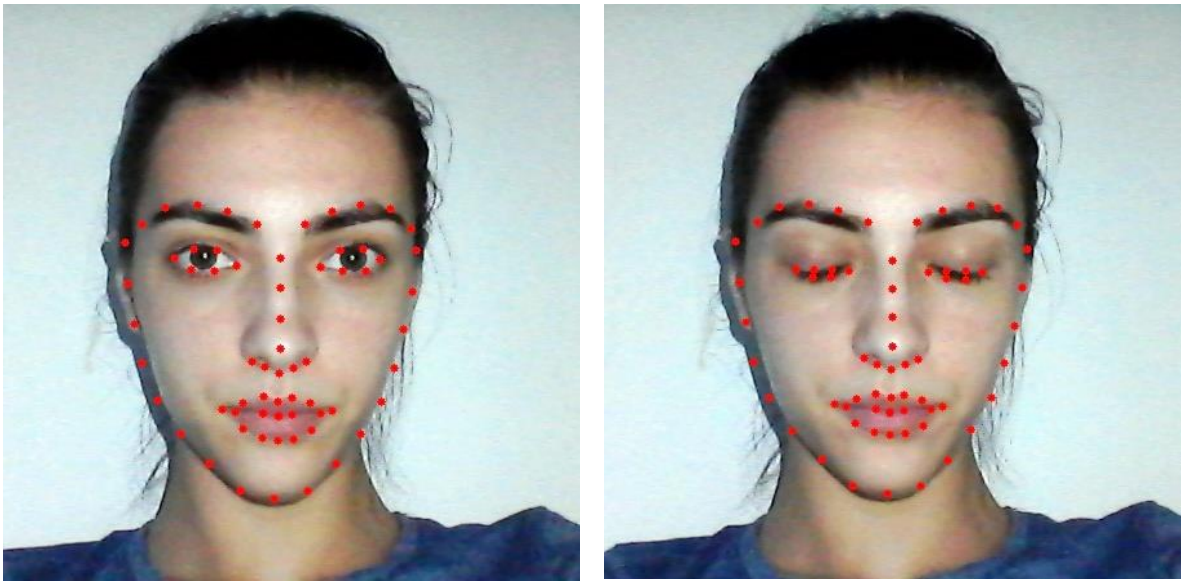


Figura2. Imagini de exemplificare cu punctele care marchează zonele importante ale feței.

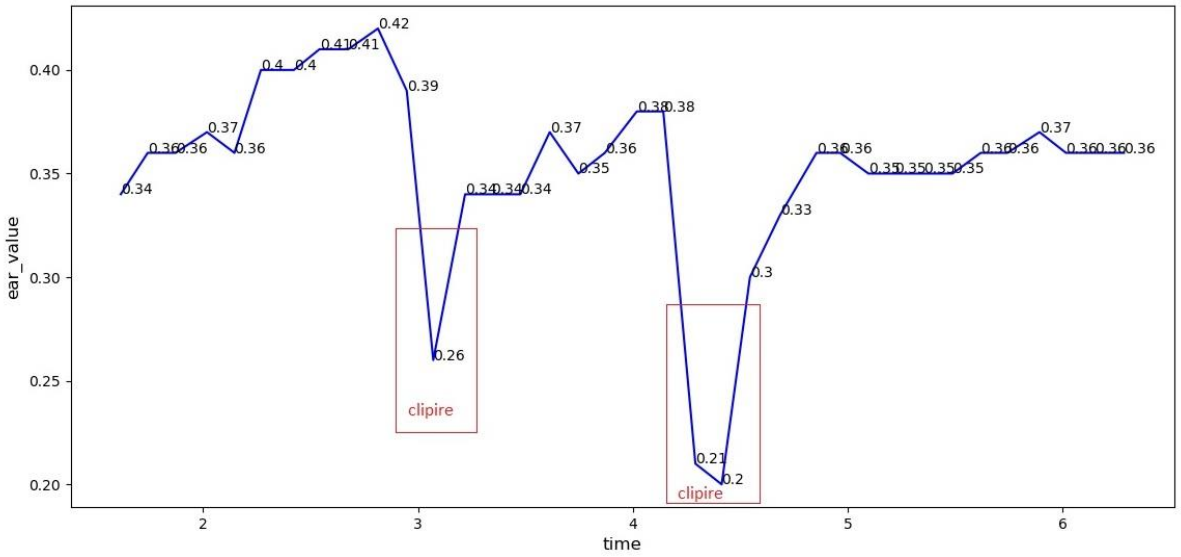


Figura3. Grafic cu valorile EAR în timp. (se pot observa 2 clipiri)

În general, dacă valoarea EAR este foarte mică, nu înseamnă neapărat că persoana clipește. O valoare scăzută poate însemna că acea persoană și-a închis intenționat ochii pentru o perioadă mai mare de timp, sau a făcut anumite grimase, expresii ale feței, căscat, strănut, etc., sau a apărut o fluctuație a punctelor returnate de algoritmul folosit pentru detectarea ochilor.

După ce s-a calculat acest raport, se folosește un algoritm de clasificare care primește ca și input frame-uri de scurtă durată din video. Experimental, s-a găsit ca pentru video-uri de 30fps[10], aproximativ 6 frame-uri pot avea impact semnificativ pentru detecția clipirilor atunci când ochiul este închis. Astfel, pentru fiecare cadru, o structură 13-dimensională caracteristică este construită prin concatenarea valorilor EAR din cele aproximativ 6 cadre vecine. Acest lucru a fost implementat cu ajutorul unui clasificator liniar SVM (numit EAR SVM) antrenat manual prin adnotarea secvențelor. În procesul de antrenare manuală, exemplele pozitive erau cadre în care se clipea, iar exemplele negative erau acele cadre din video în care nu se clipea deloc și care erau la distanță de 5-7 cadre de cadrele în care o clipire avea loc.

Pentru a testa acest algoritm, au avut loc 2 categorii de experimente. O categorie care studia acuratețea detectării punctelor necesare calculării raportului EAR, iar cealaltă categorie evalua performanța algoritmului prezentat mai sus. Conform concluziile celor care au efectuat studiul, rezultatele au fost relativ bune. Concluziile mele privind acest algoritm se gasesc în capitolul 5.2.3

[10] fps – (frames per second) reprezintă numarul de cadre pe secundă cu care pot opera unitățile de înregistrare și camerele video

4.2 Al doilea algoritm:

Cel de-al doilea algoritm pentru detectarea clipirilor este un algoritm propus de către o echipă de cercetători de la Facultatea de Inginerie, Universitatea Kasetsart din Thailanda, Departamentul de Ingineria Calculatoarele și este descris în detaliu în documentul [15]. Acest algoritm folosește în primă fază Haar Cascade Classifier și algoritmul Camshift pentru detectarea feței, iar pentru detectarea ochilor folosește un Adaptive Haar Cascade Classifier bazându-se pe proporțiile și relațiile deja cunoscute pentru poziția ochilor pe fața umană. Mai apoi, pentru a detecta clipirile, echipa propune o metodă numită ”the eyelid’s state detecting (ESD) value”. Această valoare poate fi mai apoi folosită pentru a determina dacă ochiul este închis sau deschis. Potrivit experimentelor făcute de către cercetători, acestă metodă are 99,6 % acurateţe pentru detectarea clipirilor.

Deoarece în acest capitol vom prezenta doar algoritmi pentru detectarea clipirilor, nu vom mai detalia metodele propuse de detectarea a feței și ochilor, și vom porni de la ideea că acest proces a avut loc deja, și că cunoaștem coordonatele feței și ale ochilor.

4.2.1 Descriere algoritm

Valoarea ESD[11] este un reper folosit pentru a determina starea ochiului: închis sau deschis. Valoarea poate fi calculată folosind algoritmul descris în continuare. Metoda își propune să găsească pragul (threhold) minim, astfel încât în imaginea alb-negru după ce a fost aplicat un filtru pentru blur, să existe cel puțin un pixel negru. Algoritmul primește ca și input imaginea cu ochiul. Acestei imagini i se aplică o segmentare (thresholding) care începe cu o valoare de threshold de 0, pentru a transforma imaginea într-o imagine binară (pentru fiecare pixel, există doar două valori/culori posibile, de obicei alb și negru), apoi, pentru imaginea alb-negru se aplică un filtru de blur pentru a elimina diferitele neclarități (zgomote). După aceste două operații aplicate pe imaginea inițială, se verifică dacă există măcar un pixel negru. Dacă nu există niciun pixel negru, se incrementeză valoarea de threshold și se reiau pașii descriși mai sus până se gasește acel pixel negru, moment în care algoritmul se oprește și returnează valoarea ESD.

[11] ESD - the eyelid’s state detecting value

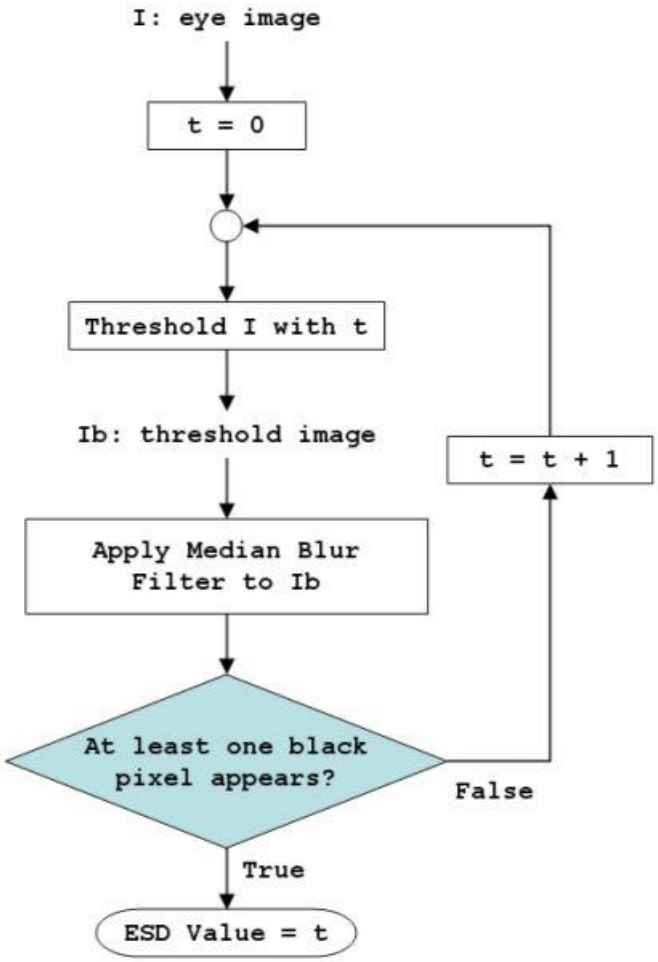


Figura4. Algoritmul propus pentru a calcula valoarea ESD

4.2.2 Threholding și segmentare de imagini.

Nu am găsit un termen în limba română care să definească acest procedeu, dar o să explic în cele ce umează ce presupune acestă operație. În contextul procesării de imagini, threholding-ul este cea mai simplă metodă de segmentare a imaginilor. În același context, segmentarea imaginilor este procesul de partiționare a unei imagini digitale în mai multe segmente (grupuri de pixeli, cunoscute și sub denumirea de ”super-pixeli”). Scopul segmentării este de simplifica sau de a schimba imaginea într-o imagine care este mai semnificativă și mai simplu de analizat. De obicei, această operație se folosește pentru localizarea și delimitarea conturului anumitor obiecte din imagini. Mai precis, segmentarea imaginilor este procesul de atribuire a unei etichete fiecărui pixel dintr-o imagine, astfel încât pixelii cu aceeși etichetă să aibă aceeași etichetă. [17]

Revenind la thresholding, această metodă se bazează pe o valoare de prag (threshold value) pentru a transforma o imagine gray-scale[12] într-o imagine binară[13]. Cheia acestei metode, este selectarea valorii potrivite pe baza căreia se face threshold. Fiecare pixel din imagine, se înlocuiește cu un pixel negru, de valoare 255, dacă intensitatea acelui pixel este mai mică decât valoarea de threshold stabilită, altfel se înlocuiește cu un pixel alb, de valoare 0.

Există mai multe categorii de metode cu ajutorul cărora se poate face thresholding. Aceste categorii sunt (Sezgin și colab., 2004):

- histograme – metodele ce fac parte din acestă cateogorie analizează curburile, punctele de minim sau maxim, sau pantele histogramei rezultate
- clusterizare – eșantioanele de nivel de culoare gri sunt grupate în două părți, ca background și foreground
- entropie – algoritmi care folosesc entropia regiunilor din foreground-ului și background
- caracteristicile obiectului – metode ce caută o similaritate între imaginea binară și cea grayscale. Ca de exemplu fuzzy shape similarity sau edge coincidence
- metode spațiale – care folosesc distribuirea probabilității de ordin mai mare și/sau corelația dintre pixeli
- metode locale – care selectează pentru fiecare pixel o valoare de threshold diferită.

Figura5. Imagine originală **Imagine după aplicarea thresholding (t=100)**

[12] grayscale image – imagine în care fiecare pixel are diferite nuanțe de alb, gri sau negru.
[13] Imagine binară – imagine în care fiecare pixel poate avea doar două culori, de obicei alb sau negru

4.2.3 Median filter

Filtrarea imaginilor se înscrie în clasa operațiilor de îmbunătățire, principalul scop al acesteia fiind eliminarea zgomotului suprapus unei imagini.

Filtrarea reprezintă o operație de vecinătate. Prin aceasta se înțelege că la calculul noii valori a unui pixel vor contribui și valorile pixelilor vecini, nu doar vechea lui valoare, cum se întâmpla la operațiile punctuale. Vecinii unui pixel reprezintă o mulțime de pixeli, aflați în apropierea acestuia, care alcătuiesc o vecinătate. Acestă vecinătate poate avea diverse forme și dimensiuni, însă cele mai utilizate în practică sunst vecinătățile de formă pătrată, de dimensiuni impare. Aceste filtre pot fi de 2 feluri:

- Filtrare liniară: filtrare ce respectă principiul liniarității. Acest pricipiu se enunță așa: Fiind date două imagini f1(x, y) și f2(x, y), și două numere reale α și β, se numește operator liniar, un operator O care are următoarea proprietate: $O[\alpha \cdot f1(x, y) + \beta \cdot f2(x, y)] = \alpha \cdot O[f1(x, y)] + \beta \cdot O[f2(x, y)]$
- Filtrare neliniară: această filtrare nu mai respectă principiul enuțat mai sus, dar a apărut din necesitatea de a depăși limitările filtrelor liniare, în ceea ce privește zgomotele care nu au o distribuție normală sau nu sunt aditive.

Figura6. Imagine originală **Imagine cu median filter**

În următorul exemplul voi detalia algoritmul pentru median filter, varianta de filtrare neliniară.

1

0	0	0	
0	**8**	**9**	**10**
0	**12**	**8**	**9**
	13	**12**	**8**

2

0	0	0
8	**9**	**10**
12	**8**	**9**
13	**12**	**8**

3

	0	0	0
8	**9**	**10**	0
12	**8**	**9**	0
13	**12**	**8**	

4

0	**8**	**9**	**10**
0	**12**	**8**	**9**
0	**13**	**12**	**8**

5

8	**9**	**10**
12	**8**	**9**
13	**12**	**8**

6

8	**9**	**10**	0
12	**8**	**9**	0
13	**12**	**8**	0

7

	8	**9**	**10**
0	**12**	**8**	**9**
0	**13**	**12**	**8**
0	0	0	

8

8	**9**	**10**
12	**8**	**9**
13	**12**	**8**
0	0	0

9

8	**9**	**10**	
12	**8**	**9**	0
13	**12**	**8**	0
	0	0	0

1

0	0	0	0	0	8	8	9	12

2

0	0	0	8	8	9	9	9	12

3

0	0	0	0	0	8	9	9	10

4

0	0	0	8	8	9	12	12	13

5

8	8	8	9	9	10	12	12	13

6

0	0	0	8	8	9	9	10	12

7

0	0	0	0	0	8	12	12	13

8

0	0	0	8	8	9	12	12	13

9

0	0	0	0	0	8	8	9	12

8	9	10
12	8	9
13	12	8

Matricea inițială

0	8	0
8	9	8
0	8	0

Matricea rezultat

Așadar, pentru fiecare element se iau vecinii acestuia (numărul de vecini este în funcție de un parametru setat inițial, de obicei acest parametru este impart, cel mai adesea se folosește 3), se ordonează, iar elementul de pe poziția din mijloc este elementul din matricea rezultat, de pe aceeași poziție.

4.2.4 Determinarea stării ochiului

După ce s-a aplicat threshold și median blur, se aplică algoritmul din figura 4 și se află t, adică valoarea ESD. După ce se știe acestă valoare, se definește un automat finit pentru a putea determina starea în care se află ochii. Automatul este descris în figura de mai jos.

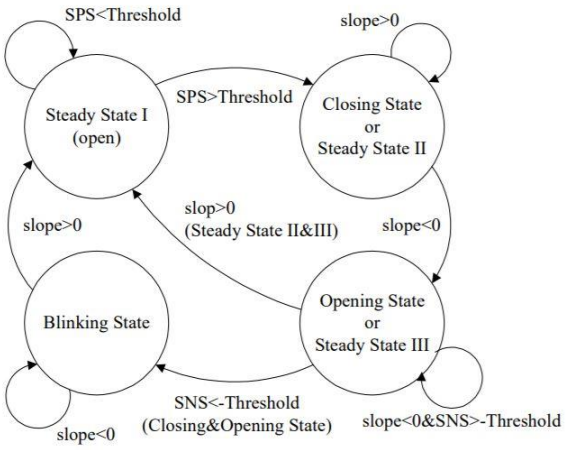
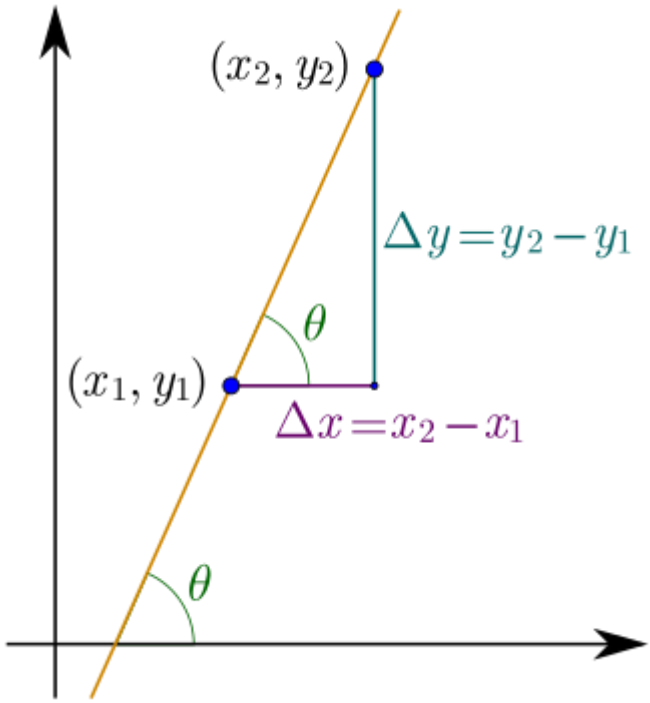


Figura7. Automat pentru determinarea stării ochilor

Automatul are 4 stări, după cum se poate observa și în imagine, iar trecerea între stări se face folosid 3 variabile de decizie care sunt calculate în funcție de graficul cu valori ESD: panta (slope), SPS[14] și SNS[15]. Panta este panta de la momentul curent, determinată de ultimele 2 puncte, valoarea SNS reprezintă suma pantelor negative consecutive, iar SPS reprezintă suma pantelor pozitive consecutive. Procesul începe din prima stare, când ochiul este deschis. Acestă stare este schimbată în starea a2a, de ochi închiși dacă SPS este mai mare decât o valoarea threshold. Din acestă stare se poate trece în a treia stare dacă panta din acel moment este negativă. Apoi, din starea a treia se trece înapoi în starea inițială dacă valoarea pantei este mai

[14] SPS – Sum of Positive Slope
[15] SNS – Sum of Negative Slope

mare ca zero, sau se poate trece în starea 4, de clipire, dacă SNS este mai mică decât valoarea negativă de threshold.



Panta unei drepte este definită de relația:

$$m = \frac{\Delta y}{\Delta x}$$

Dacă m>0, spunem ca panta este pozitivă, iar daca m<0, panta este negativă

Figura8. Reprezentare pantă

Capitolul 5. Implementarea algoritmilor descriși

În acest capitol voi detalia metodele pe care le-am implementat pentru a detecta clipirile. Pentru început voi prezenta pașii pentru pregătirea mediului de lucru.

5.1 Pregătirea mediului de lucru

- Pentru tot ce urmează a fi prezentat am folosit Python 3.6 pe Windows 10.
- Am instalat Python 3.6;
- Am descărcat, instalat și adăugat ca și environment variable Anaconda. Anaconda este o distribuție de Python și R care include ca și limbaj de bază Python, vine cu peste 100 de pachete python preinstalate (numpy, scipy, etc). De asemenea are și un IDE/editor –Spyder, și propriul package manager, conda, folosit pentru a instala, updata diferite pachete. Este disponibilă pentru Windows, Linux. [16]
- `conda create --name opencv-env python=3.6`
- `activate opencv-env`
- Am instalat librăria OpenCV cu comanda:
  `python -m pip install opencv-python`
- Am instalat librăria Dlib:
  `Pip install https://pypi.python.org/packages/da/06/bd3e241c4eb0a662914b3b4875fc52dd176a9db0d4a2c915ac2ad8800e9e/dlib-19.7.0-cp36-cp36m-win_amd64.whl` [16]

5.1.1 Despre OpenCV

OpenCV ”Open Source Computer Vision Library” este o bibliotecă de funcții opensource, scrisă în C și C++. Librăria a fost proiectată astfel încât să ofere eficiență computațională pentru aplicații de computer vision în timp real. Biblioteca conține peste 500 de funcții care acoperă domenii precum imagistică medicală, securitate, calibrarea camerelor foto, vedere stereo sau robotică. OpenCV mai conține de asemenea un set complet de funcții de machine learning. Scopul librăriei este de a pune la dispoziție utilizatorilor o infrastuctură de procesare a imaginilor ce poate fi folosită în dezvoltarea rapidă a unor aplicații complexe. Este oferită gratuit celor care doresc să o utilizeze și este open source sub licență BSD. [18]

[16] https://anaconda.org/anaconda/python

5.1.2 Despre DLIB

Dlib este o librărie open source scrisă în C++. Davis King este autorul principal al acestei librării, librărie la care a început să se lucreze din 2002. Între timp, Dlib a crescut foarte mult, adăugându-se o largă varietate de tool-uri. În acest moment, librăria conține componente software pentru rezolvarea sistemelor de rețea, thread-uri, interfețe grafice, structuri complexe de date, algebră liniară, învățare automată, prelucrare de imagini, data mining, parsare de text, optimizare numerică, etc.

5.2 Implementare

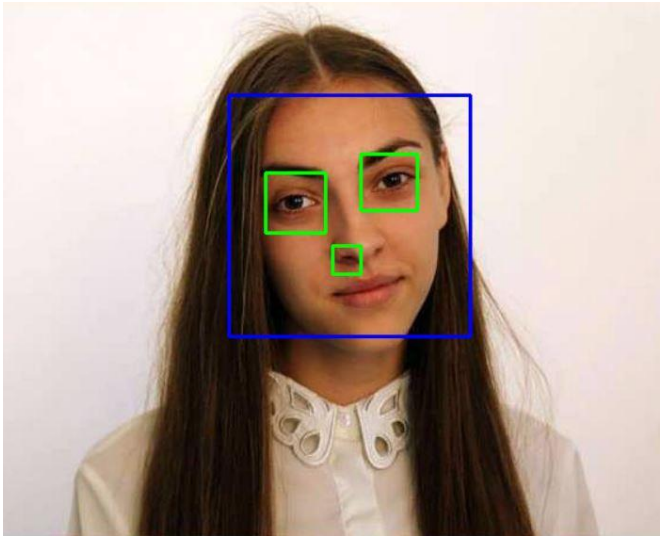
Am început implementarea cu detecția feței și a ochilor, urmând să mă axez apoi pe găsirea clipirilor. Pentru început am încercat detectarea feței și ochilor cu ajutorul OpenCV.

```
face_cascade = cv2.CascadeClassifier('C:\\Users\\AnaMariaBodnar\\Anaconda3\\Lib\\site-packages\\cv2\\data\\haarcascade_frontalface_default.x
eye_cascade = cv2.CascadeClassifier('C:\\Users\\AnaMariaBodnar\\Anaconda3\\Lib\\site-packages\\cv2\\data\\haarcascade_eye.xml')

img = cv2.imread('face1.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray,
    scaleFactor=1.1,
    minNeighbors=1,
    minSize=(10, 10))
for (x,y,w,h) in faces:
    img = cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)
    roi_gray = gray[y:y+h, x:x+w]
    roi_color = img[y:y+h, x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray)
    for (ex,ey,ew,eh) in eyes:
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)
```

Mai sus se găsește codul folosit pentru detectarea feței și a ochilor. Dar această abordare nu a dat rezultate foarte bune, oricât aș fi modificat parametrii funcției `detectMultiScale()`.

Pentru detectarea feței rezultatele erau relativ bune, dar pentru detectarea ochilor, nu erau deloc bune. De cele mai multe ori, îmi detecta cel puțin 3 ochi, într-o poză cu o singură pesoană, adică o față și 2 ochi.



Chenarul albastru reprezintă forma feței, iar cele verzi ar trebuie să marcheze ochii. Se poate observa că OpenCV a găsit un ochi în plus.

Figura9. Rezultate OpenCV pentru detectarea feței și a ochilor

Având în vedere rezultatele obținute, a fost nevoie de căutarea altei metode mai eficiente pentru a putea detecta fața și în special ochii. În acest sens, am găsit o metodă care dă rezultate foarte bune, fără a prezenta erori în niciun test pe care l-am făcut. Este vorba despre facial landmarks descris mai jos. Am ales să folosesc detectorul inclus în librăria dlib pentru implementare.

5.2.1 Detectarea feței

Detectarea punctelor cheie (facial landmarks) este un subgrup al problemei de predicției a forme (shape prediction problem). Având o imagine de intrare (și în mod normal un ROI care specifică obiectul de interes), un predictor de formă încearcă să localizeze punctele cheie de interes de-a lungul formei. În contextul reperelor faciale, scopul nostru este de a detecta structuri faciale importante din cadrul feței folosind metode de predicție a formei. După cum precizam și mai sus, pentru a detecta organe importante din cadrul feței, trebuie pentru început să localizăm fața.

Având coordonatele feței calculate la primul pas, putem să le folosim pentru cel de-al doilea pas, și anume detectarea diferitelor puncte importante din cadrul feței, cum ar fi ochii(ochiul stâng, ochiul drept, sprânceana dreaptă, sprânceana stângă), gura, nasul, etc. [12]

Detectorul facial landmark inclus în librăria dlib este implementat în *One Millisecond Face Alignment with an Ensemble of Regression Trees*[17] de către Kazemi și Sullivan în 2014.

[17] https://www.semanticscholar.org/paper/One-millisecond-face-alignment-with-an-ensemble-of-Kazemi-Sullivan/1824b1ccace464ba275ccc86619feaa89018c0ad

Acestă metodă se bazează pe 2 pași. În primul pas, este un set de antrenament cu imagini care sunt adnotate manual cu coordonatele (x, y) pentru fiecare parte a feței (ochi, nas, gură, etc), iar al doilea pas ia în considerare probabilitate de distață dintre perechile de pixeli primiți ca date de intrare. Având datele de antrenament, sunt construiți mai mulți arbori de regresie pentru a estima pozițiile fiecărei parți a feței, adică nu are loc o extragere a caracteristicilor). Rezultatul final este un detector pentru părțile feței care poate fi folosit cu succes pentru detectarea ochilor, și nu numai, în timp real cu o precizie foarte mare.

Acest detector din dlib estimează locația a 68 de puncte de coordonate (x,y) dispuse pe toată suprafața feței, în zonele importante. În imaginea de mai jos, se pot observa cele 68 de puncte, puncte ce se bazează pe baza de date IBUG 300-W[18] [12]

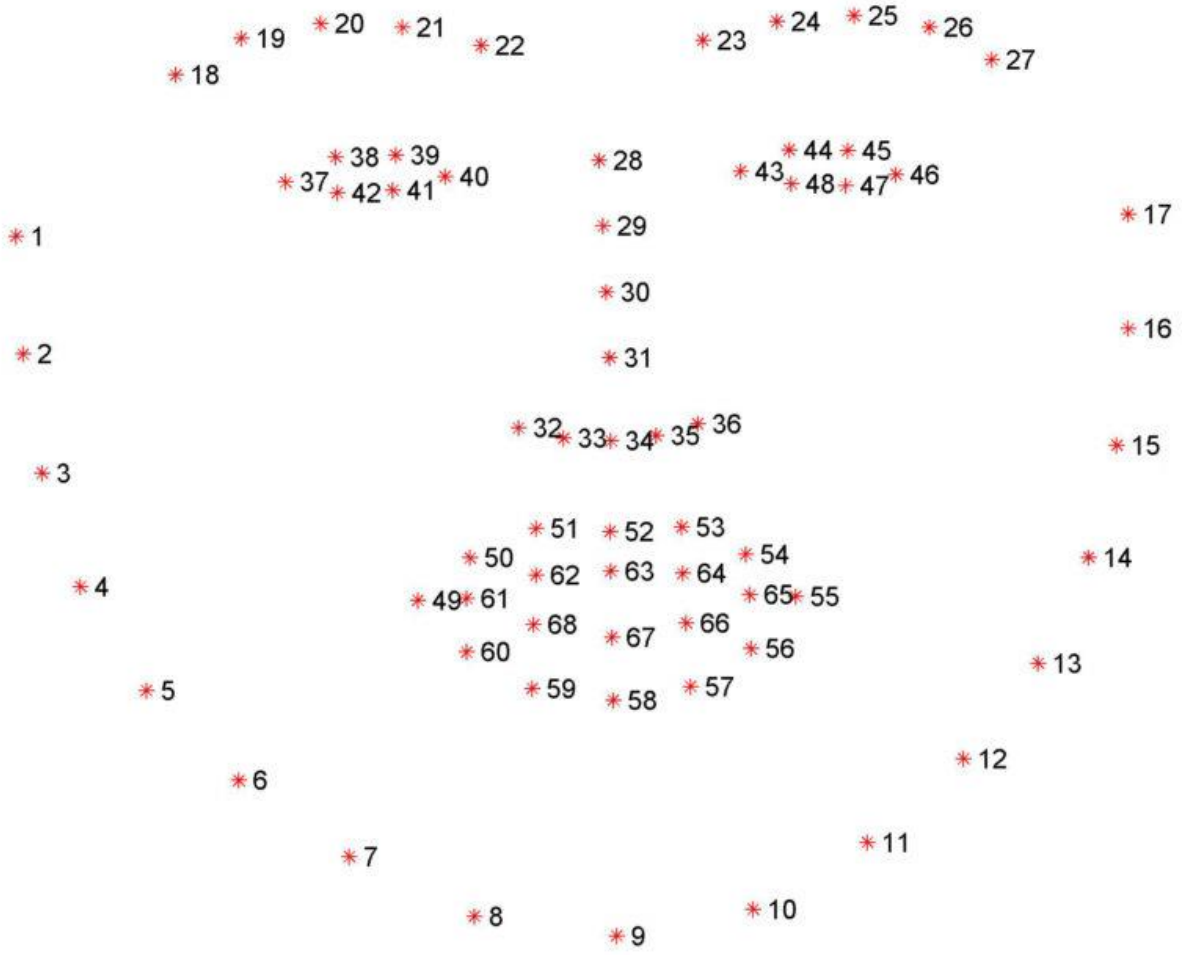


Figura10. Distribuirea celor 68 de puncte pe suprafața feței

[18] https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/

5.2.2 Implementarea primului algoritm

Atât pentru acest algoritm, cât și pentru următorul implementat, am folosit metoda descrisă mai sus pentru a obține fața și ochii.

Pentru început am definit funcția care calculează valoarea EAR pentru coordonatele unui ochi dat ca parametru, valoare dată de formula1.

```
def ear(eye):
    a= numpy.linalg.norm(eye[1] - eye[5])
    b= numpy.linalg.norm(eye[2] - eye[4])
    c= numpy.linalg.norm(eye[0] - eye[3])
    r=(a+b)/(2.0*c)
    return r
```

Apoi am obținut coordonatele feței și ale ochilor cu ajutorul dlib.

```
predictor_path="C:\\Users\\User\\Anaconda3\\Lib\\site-packages\\dlib\\shape_predictor_68_face_landmarks

'''inițializare detectorul pentru față din dlib și
creearea predictorului pentru celelalte părți ale feței'''

predictor = dlib.shape_predictor(predictor_path)
detector = dlib.get_frontal_face_detector()

#obțin indecși cei 2 ochi
(l_start, l_end) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]
(r_start, r_end) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]

#incep stream-ul video
cap=cv2.VideoCapture(0)

while True:
    #extrag frame cu frame și-l convertesc la grayscale
    src, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    #obtin fețele
    faces=detector(gray, 0)

    #parcurg lista cu fețele obtinute
    for rect in faces:
        #determin punctele cheie și convertesc coordonatele în array-uri numpy
        shape=predictor(gray, rect)
        shape=face_utils.shape_to_np(shape)

        #extrag și coordonatele ochilor
        left_eye=shape[l_start:l_end]
        right_eye=shape[r_start:r_end]

        l=numpy.array(left_eye)
        r= numpy.array(right_eye)

        #calculez raportul ear cu ajutorul coordonatelor obtinute
        right_eye_ear=ear(r)
        left_eye_ear=ear(l)

        average_ear=(right_eye_ear+left_eye_ear)/2.0
```

După ce am obținut poziția ochilor, calculăm raportul EAR, de care ne vom folosi pentru a ne da seama dacă ochii sunt închiși sau deschiși. Acest raport este mai apoi comparat cu o valoare de limită, valoare pe care am determinat-o în urma multiplelor teste. Evident că această valoare este aproximativă, este o valoare variabilă. Mai jos este restul codului cu explicațiile pentru fiecare pas.

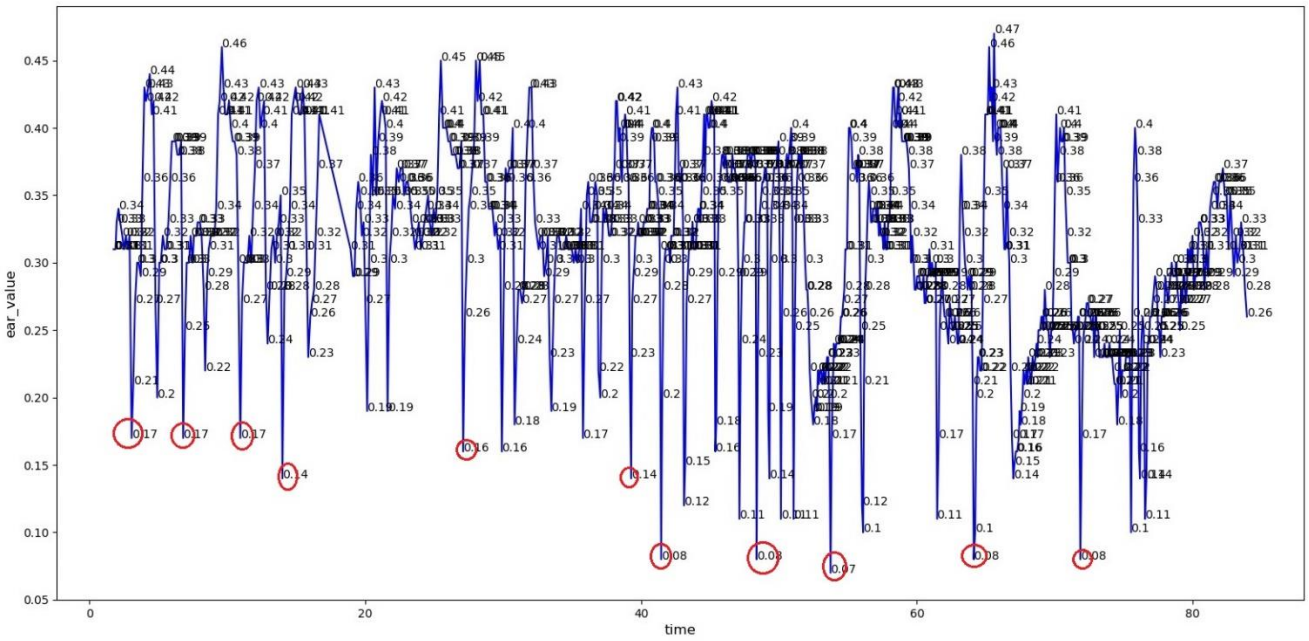
```
88
89          #desenez conturul ochilor pe frame
90          left_eye_ellipse=cv2.fitEllipse(left_eye)
91          right_eye_ellipse=cv2.fitEllipse(right_eye)
92
93          #verific daca valoarea ear calulata este mai mica decat limita
94          # si incrementez un counter daca se intampla asta
95          if average_ear<ear_thresh:
96              count+=1
97          #altfel, valoarea ear nu indică vreun indiciu de clipire
98          else:
99              #dacă ochii sunt închiși pentru suficient timp,
100             #incrementez numarul total de blink-uri
101             if count >= 3:
102                 blinks += 1
103
104             #resetez counterul de frame
105             count=0
106
107             #afisez numarul total de clipiri
108             cv2.putText(frame, "Blinks: {}".format(blinks), (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
109         for(x, y) in shape:
110             cv2.circle(frame, (x, y), 3, (0, 0, 255), -1)
111         #salvez fiecare frame ca si imagine jpg pentru a analiza rezultatele
112         cv2.imwrite(pathOut + "/%#05d.jpg" % (count_frame), frame)
113         count_frame += 1
114
115     cv2.imshow('Video', frame)
116     if cv2.waitKey(1) & 0xFF == ord('q'):
117         break
```

După ce am reușit să determinăm clipirile, am testat programul scris pe o bază de date proprie, cu 20 de filmulețe, cu o durată de la 5 secunde până la 30 de secunde. O să prezint în tabelul de mai jos, numărul adevărat de clipiri (numărate manual) și numărul returnat de către program, pentru 10 dintre filmulețe.

Video id	Numărul real de clipiri	Numărul de clipiri returnat de program
1	4	4
2	15	8
3	10	2
4	5	4
5	2	2
6	13	11
7	1	0
8	18	15
9	3	0
10	5	4

Tabel 2. Eficiența primului algoritm

Atașez un grafic care este sugestiv despre cum se modifică valoarele ear pe parcursul unui video în care o persoană clipește timp de 17 secunde.



5.2.3 Concluzii

După cum se poate observa din tabelul 2, această metodă este destul de instabilă. Ceea ce am observat eu, este că metoda dă rezultate bune dacă clipirile sunt “bine definite”, adică dacă se clipește clar, nu foarte rapid, iar atunci când nu se clipește, adică ochii sunt deschiși, aceștia trebuie să fie larg deschiși. Dacă ochii nu sunt larg deschiși, programul consideră că sunt închiși, iar în momentul când o să fie deschiși mai tare, o să se considere că tocmai a avut loc o cîmpie.

Este evident că acea valoare de limită, prefixată, cu care compar valoarea EAR calculată pentru fiecare frame, nu este cea mai bună variantă, pentru că, pentru fiecare persoană, valoarea aceea variază. Pentru unii, pleoapele pot acoperi mai mult partea albă a ochiului atunci când acesta este deschis, sau efectiv forma ochiului să fie mai oblică (adică se apropie mai mult de un ochi închis) cum este de exemplu la persoanele de origine asiatică, caz în care metoda propusă este total ineficientă. De asemenea, dacă persoana zâmbește, ochii tind să pară a fi închiși, dar nu sunt în realitate, lucru de care algoritmul propus nu-și dă seama.

Deși acest algoritm este foarte simplu, și din testele făcute nu pare a avea cele mai bune rezultate implementat de unul singur, cred că raportul pe care se bazează algoritmul poate fi folosit cu succes în cadrul unui alt algoritm în condițiile în care este făcută mai atent și detaliat trierea în funcție de acea valoare calculată.

5.2.4 Implemetarea celui de-al doilea algoritm

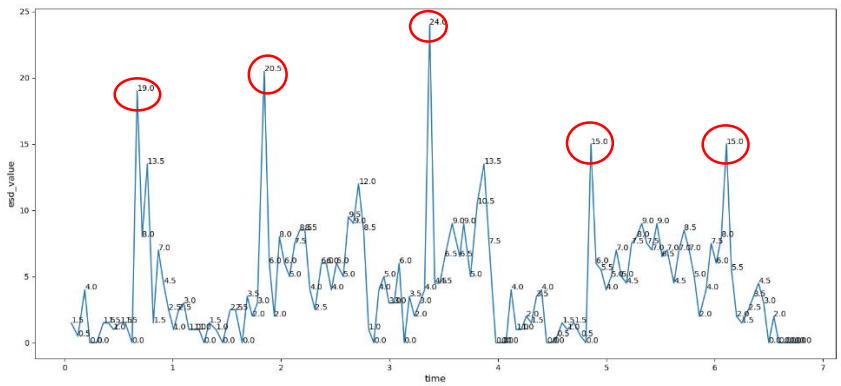
Pentru că primul algoritm nu a dat rezultate foarte bune, am trecut la implemetarea unui alt algoritm, și anume algoritmul prezentat în capitolul 4.2.

Algoritmul inițial folosește Haar Cascade Classifier și algoritmul Camshift pentru detectarea feței, iar pentru detectarea ochilor și de asemenea folosește un Haar Cascade Classifier ușor modificat pentru detectarea ochilor. Pentru că metoda folosită în algoritmul precedent de detectare a feței și ochilor a dat rezultate foarte bune, am decis să o folosesc și aici, preluînd din algoritmul din capitolul 4.2 doar metoda pentru detectarea clipirilor.

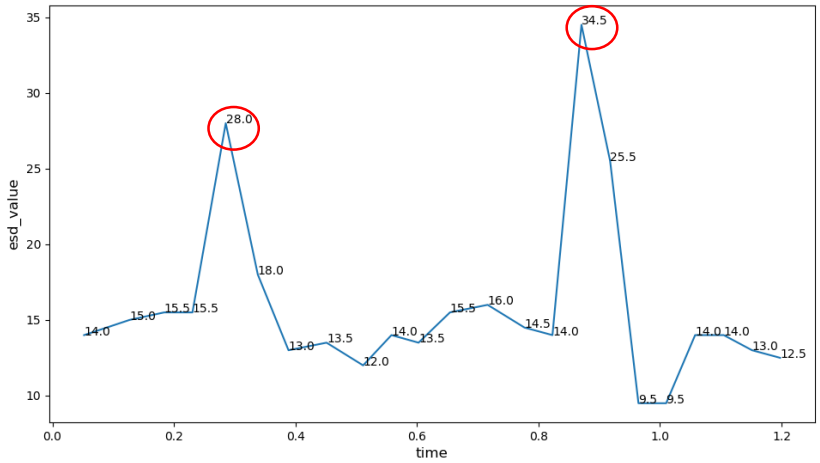
Așadar, am urmat pașii de mai sus pentru obținerea coordonatelor pentru față și ochi, iar odată ce am obținut aceste puncte, am aplicat algoritmul descris în figura 4 pentru a căuta valoare ESD care mă va ajuta mai târziu.

```
def search_esd_value(image):
    #inițial pornim cu t=0
    t=0
    found = False
    while not found:
        #aplic thresholding pe frame-ul primit ca parametru
        rec, thresh=cv2.threshold(image, t ,255, cv2.THRESH_BINARY)
        #aplic median blur peste imaginea binară
        median_blur=cv2.medianBlur(thresh, 3)
        #convertesc lista de liste median_blur într-o listă simplă
        dmedian_blur = median_blur.ravel()
        #sortez lista obtinută
        sorted_v=np.sort(dmedian_blur)
        #daca primul element este 0, am un pixel negru si pot returna t
        if sorted_v[0] == 0:
            found = True
        else:
            t += 1
    return t
```

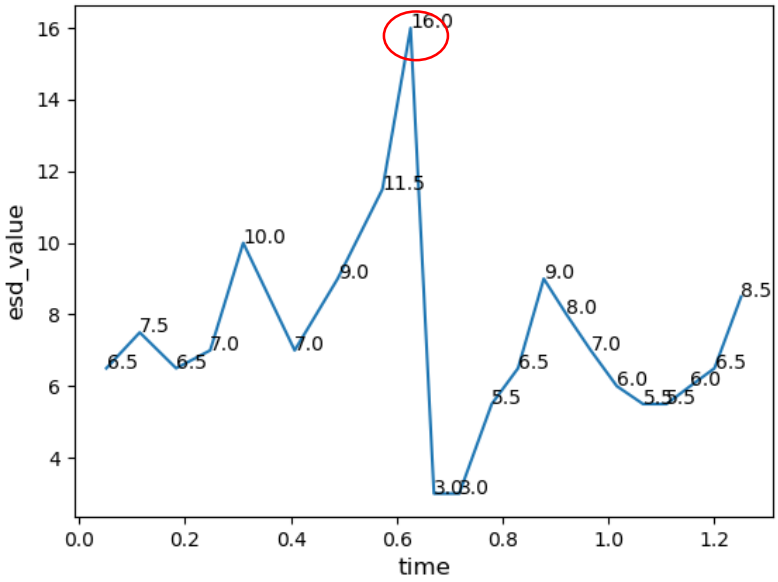
Atașez mai jos câteva grafice cu valorile esd calculate de funcția de mai sus pe parcursul câtorva filmulețe din baza de date folosită pentru acest proiect.



5 clipiri



2 clipiri



1 clipire

După câte se poate observa în graficele de mai sus, marcările cu roșu sunt momentele în care se produc clipirile, dar se mai observă că valorile care mă ajută să construiesc aceste grafice, valorile returnate de funcția de mai sus, nu sunt constante, nu se încadrează într-un anumit interval, deci nu mă pot folosi directe de aceste valori pentru a încerca să trag diferite concluzii. Din ceea ce am observat, această iregularite a valorilor depinde în mare parte de calitatea imaginii. Dacă video-ul are luminozitatea mică, automat valorile voi fi mai mici, dar dacă condițiile de lumină sunt relativ normale, valorile esd atunci când are loc o clipire se apropie de 20, 30, 40 chiar 50, însă și aici, intervalul este destul de mare.

Revenind la restul programului, pe parcursul înregistrării, după ce pentru fiecare frame este calculată această valoare, se calculează valorile pentru pantă, suma pantelor pozitive consecutive și suma pantelor negative consecutive pentru a putea determina dacă ochiul este închis sau deschis.

Deoarece codul pentru detectarea feței este același cod prezentat și la metoda anterioară, o să prezint direct codul pentru determinarea clipirilor din bucla while care determină coordonatele feței și ale ochilor.

```
            s = ''
            if sps <= threshold_value:
                flag_open = True
                s = "stare 1 (open)"
            if sps > threshold_value or sns < (threshold_value * -1):
                s = "stare 2 (close)"
                flag_close = True
            if sns < (threshold_value * -1) and flag_close is True:
                s = "stare 3 (blink)"
                flag_3 = True
            elif sns < (threshold_value * -1) and flag_3 is True:
                flag_blink = True
                s = "stare 4 (blink)"


            if (s == "stare 1 (open)" and s1=="stare 2 (close)" and s2=="stare 3 (blink)")
            or (s=="stare 1 (open)" and s1=="stare 3 (blink)"):
                count_blinks+=1
                if count_blinks%2==1:
                    flag_start_blink=True
                    flag_stop_blink=False
                else:
                    flag_stop_blink=True
                    flag_start_blink=False
                    cv2.putText(frame, "blinks: " + str(count_blinks // 2), (500, 110), cv2.FONT_
                    blinks=count_blinks//2

            if flag_start_blink is True:
                count_close+=1
                print(count_close)
            if flag_stop_blink is True:
                count_close=0
            s1=s
            s2=s1
```

La fel ca și pentru primul algoritm, am testat acest program pe aceeași bază de date. Rezultatele se găsesc în tabelul de mai jos.

Video id	Numărul real de clipiri	Numărul de clipiri returnat de program
1	4	4
2	15	14
3	10	8
4	5	4
5	2	2
6	13	12
7	1	1
8	18	15
9	3	4
10	5	4

Tabel 3. Eficiența celui de-al 2lea algoritm

Este evident că rezultatele sunt mult mai bune. Având certitudinea că am o metodă cu rezultate bune în detectarea clipirilor, se poate trece la pasul următor, și anume la a încerca să-mi dau seama când persoana este pe punctul de a adormi.

Nu este deloc ușor să faci un computer să-și dea seama când o persoană este obosită și pe punctul de a adormi, având în vedere că decizia pe care trebuie să o ia depinde de foarte mulți parametri care variază de la persoană la persoană. Am ales să măsor durata clipirilor, iar dacă persoană ținea ochii închiși mai mult timp, se pornește o alarmă.

Concluzii

Am învățat foarte multe din acestă lucrare care se presupune a fi cea mai importantă lucrare făcută de mine pe tot parcursul facultății. Am devenit mai conștientă de faptul că, deși oamenii de știință pot lucra foarte mult pe un anumit subiect, rezultatele muncii lor ar putea fi în continuare nesatisfăcătoare. Am mai aflat că este esențial să realizăm cercetarea într-un mod adecvat. Dacă cineva aspiră să facă niște cercetări în domeniu, și nu este prea familiar cu acesta, este necesar să citească mult înainte de a se apuca de treabă. Inițial, mă gândisem ca proiectul meu să studieze clipirile și să-și dea seama de oamenii care mint. La această idee a contribuit și doctor House, personajul principal din serialul cu același nume, care spunea mereu că ”Toată lumea minte.”. Am studiat puțin acest subiect și mi-am dat seama că mi-ar fi fost extrem de dificil, în primul rând să colectez baza de date cu videoclipuri cu oameni care mint, iar apoi să separ adevărul de minciună. Am decis să nu schimb în totalitate subiectul și să studiez în continuare detectarea clipilor, însă cu o aplicabilitare mai simplu de abordat: somnolența în trafic.

Consider că fiecare pas făcut pentru rezolvarea aceste probleme este important și trebuie făcut cu foarte mare atenție. De la studierea problemei, documentarea, colectarea bazei de date pentru testare, implementarea algoritmilor, până la tragerea concluziilor.

Ceea ce am reușit să fac în principal, și anume detectarea clipirilor, poate fi folosit cu succes în anticiparea oboselii la volan, dar poate fi utilizat și în alte proiecte, cum ar fi interacțiunea persoanelor cu dizabilități cu computerul sau chiar detectarea minciunilor despre care vorbeam mai sus.

Bibliografie

[1] Eye Blinking Detection and Analysis, Tomasz Jarzski http://tomaszjarzynski.pl/wp-content/uploads/2016/01/book.pdf

[2] Eye-Blink Detection Using Facial Landmarks Tereza Soukupova https://dspace.cvut.cz/bitstream/handle/10467/64839/F3-DP-2016-Soukupova-Tereza-SOUKUPOVA_DP_2016.pdf

[3] https://www.reginamaria.ro/articole-medicale/de-ce-clipim

[4] Grace, R & Byrne, VE & Bierman, DM et al: A Drowsy Driver Detection System for Heavy Vehicles. Proceedings of the 17th Digital Avionics Systems Conference, Vol. 2, 2001, pp. I36/1 – I36/8.

[5] Vehicle control and drowsiness – Albert Kircher, Marcus Uddman, Jesper Sandin

http://www.diva-portal.org/smash/get/diva2:673709/FULLTEXT01.pdfVTI

[6] Wierwille, WW & Ellsworth, LA & Wreggit, SS & Fairbanks, RJ & Kirn CL: Research on Vehicle-Based Driver Status/Performance Monitoring: Development, Validation, and Refinement of Algorithms for Detection of Driver Drowsiness. National Highway Traffic Safety Administration Final Report: DOT HS 808 247, 1994. (October 1998. Publication No. FHWA-MCRT-98-006)

[7] PERCLOS: A Valid Psychophysiological Measure of Alertness as Assessed by Psychomotor Vigilance – TechBrief FHWA-MCRT-98-006, October 1998.

[8] DRIVER DROWSINESS DETECTION BASED ON EYE BLINK, INDRACHAPA BUWANEKA BANDARA http://bucks.collections.crest.ac.uk/9782/1/Bandara%20Indrachapa%20-%20Thesis%20pdf.pdf

[9] Real-Time Eye Blink Detection using Facial Landmarks, Tereza Soukupova and JanCech

http://vision.fe.uni-lj.si/cvww2016/proceedings/papers/05.pdf

[10] http://tomaszjarzynski.pl/wp-content/uploads/2016/01/book.pdf

[11] Fast computation of PERCLOS and Saccadic Ratio, Anirban Dasgupta https://pdfs.semanticscholar.org/22f4/e796de8f7fc9c1074af38ebf7d1d9beb0204.pdf

[12] https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/

[14] Q. Ji, Z. Zhu, and P. Lan, “Real-time nonintrusive monitoring and prediction of driver fatigue,” IEEE Transactions on Vehicular Technology, vol. 53, pp. 1052–1068, July 2004. 9, 10

[15] A Method for Real-Time Eye Blink Detection and Its Application Chinnawat Devahasdin Na Ayudhya https://pdfs.semanticscholar.org/6a16/b91b2db0a3164f62bfd956530a4206b23fea.pdf

[16] https://www.learnopencv.com/install-opencv-3-and-dlib-on-windows-python-only/

[17] Despre segmentare http://www.miv.ro/ro/documentatie/pi/PIlab10.pdf

[18] Despre OpenCV http://imag.pub.ro/common/staff/cflorea/papers/SSPI_OpenCV.pdf

[19] http://dlib.net/intro.html