Petit Projecte de Visió per Computador

Detecció de mirades

Francesc Forn Cañabate
Raúl Lumbreras Navarro
Martí Vall Oliveira

Índex

Introducció

L'objectiu del projecte és implementar un sistema automàtic per detectar la mirada mitjançant visió per computador. El sistema ha de ser capaç de detectar on són els ulls i posteriorment mesurar l'angle horitzontal de la mirada en base a la posició del iris. Les imatges amb les que es treballaran seran amb un enquadrament tipus bust d'un usuari sense ulleres en una orientació vertical amb rotacions del cap poc apreciables; tal com es mostra a continuació en la imatge de la figura següent:

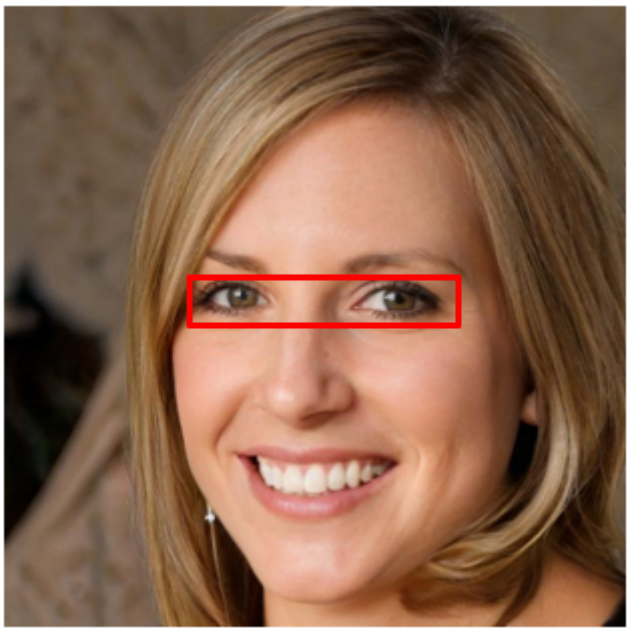

Figura 1. Imatge d'exemple d'una detecció d'ulls (imatge extreta de thispersondoesnotexist.com).

Idea inicial (Checkpoint)

Base de dades

Per tal d'entrenar els classificadors, hem emprat un conjunt d'imatges de persones generades per IA de la web *thispersondoesnotexist.com* passades a blanc i negre. Cada una d'aquestes imatges realistes l'hem segmentat en imatges de 512x128, mida suficient pel rectangle que conté els ulls en la majoria d'imatges. Aquestes imatges o bé correspondran als ulls de la persona, o bé a qualsevol altra part de la foto. Un exemple de cada es troben en les següents figures:

Figura 2: Imatge de característiques diferents a ulls.

Figura 3: Imatge d'ulls.

Per als passos d'entrenament i testeig del model, la base de dades ha quedat dividida de la següent manera:

	Ulls	No-Ulls
Entrenament	162	6999
Testeig	30	30

Descriptors i Classificadors

Com a primera idea de realització del projecte, vam optar per classificar els ulls usant sels següents descriptors:
- l'histograma de grisos.
- l'histograma de gradient (HOG).

Pel que fa als classificadors, els que vam emprar:
- Fine tree
- SVM (Support Vector Machine)
- NN (Neural Network)

Histograma de grisos

La primera estratègia que vam considerar va ser emprar la informació de l'histograma de grisos de les imatges d'entrenament, però fent servir el Classification Learner vam veure que l'histograma de grisos per si sol no es un bon descriptor. A continuació veiem les matrius de confusió resultat d'entrenar diferents classificadors amb les dades d'entrenament i fent k-fold validation amb k = 5.

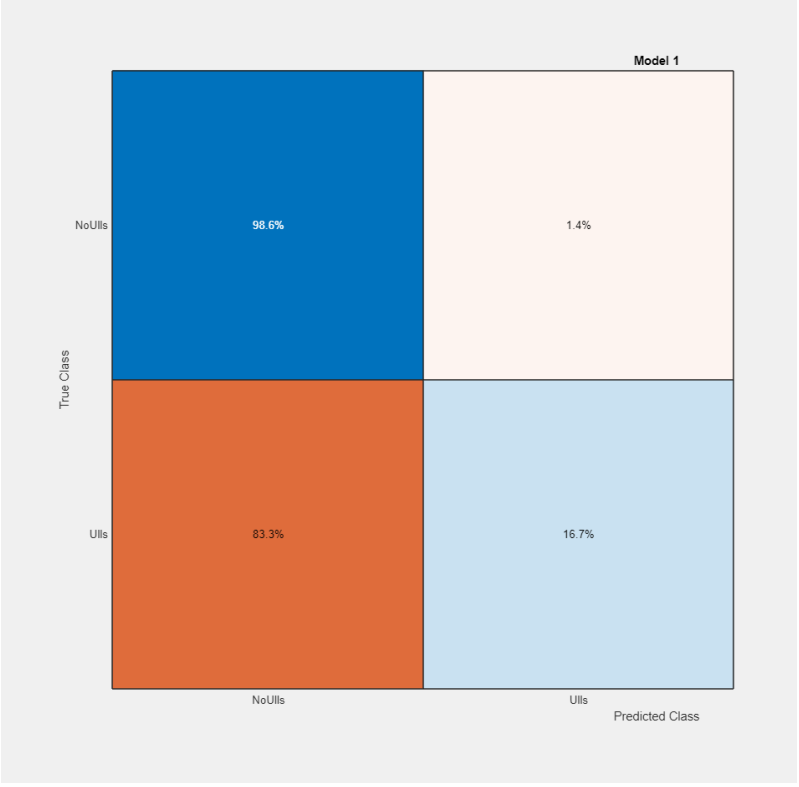

Figura 4: Matriu de confusió generada pel Classification Learner utilitzant Fine tree.

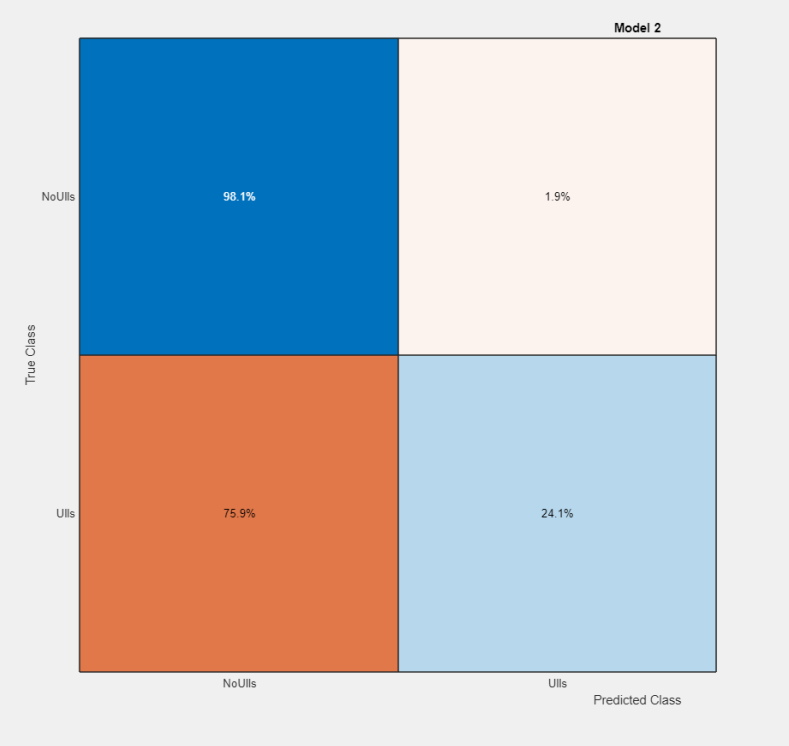

Figura 5: Matriu de confusió generada pel Classification Learner utilitzant Neural Network.

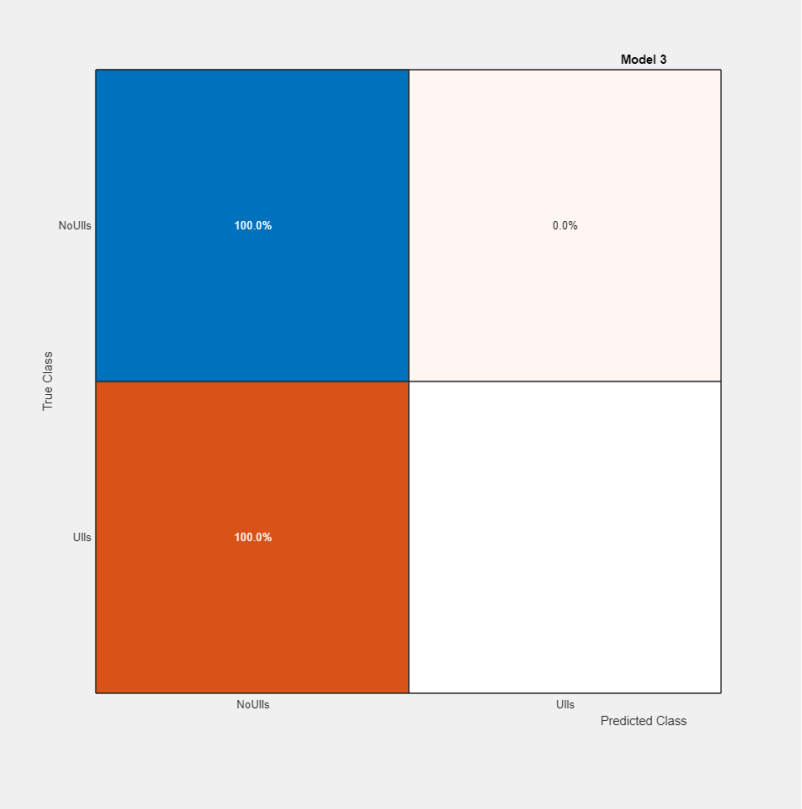

Figura 6: Matriu de confusió generada pel Classification Learner utilitzant SVM.

Va resultar clar que usar histogrames de gris per a detectar ulls no és la millor estratègia. A més, donats els resultats obtinguts amb el Classification Learner, ens vam adonat que per a les SVM, a part de obtenir uns resultats molt més pobres que amb els altres dos classificadors, també tenien un temps d'execució ridículament elevat. Per aquesta raó , en el seu moment vam decidir obviar aquest classificador i centrar-nos en els altres dos. Entre els altres dos també vam observat que d'entrada la Neural Network donava bastant millors resultats que el Fine Tree.

Detecció d'ulls amb HOGs

La segona estratègia que vam emprar va ser l'ús de HOGs. Amb HOGs es reconeixen de manera clara tant l'iris com els llagrimals, fent d'aquests descriptors idonis per a la tasca que es vol realitzar:

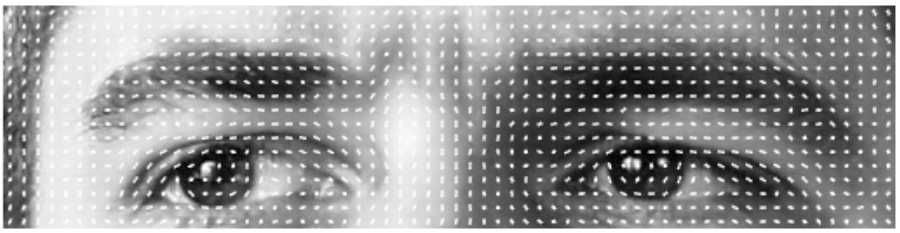

Figura 7: Imatge d'ulls amb els gradients graficats.

Per al set de testeig, vam fer proves amb treebagger i xarxes neuronals.

TreeBagger

Nom Fitxer	Valor real	Valor predit	Score NoUlls	Score Ulls
"1.jpg"	"NoUlls"	"NoUlls"	"1"	"0"
"1e.jpg"	"Ulls"	"NoUlls"	"0.84"	"0.16"
"2.jpg"	"NoUlls"	"NoUlls"	"0.93"	"0.07"
"2e.jpg"	"Ulls"	"NoUlls"	"0.81"	"0.19"
"3.jpg"	"NoUlls"	"NoUlls"	"1"	"0"
"3e.jpg"	"Ulls"	"Ulls"	"0.48"	"0.52"
"4.jpg"	"NoUlls"	"NoUlls"	"1"	"0"
"4e.jpg"	"Ulls"	"Ulls"	"0.27"	"0.73"
"5.jpg"	"NoUlls"	"NoUlls"	"1"	"0"
"5e.jpg"	"Ulls"	"Ulls"	"0.31"	"0.69"
"6.jpg"	"NoUlls"	"NoUlls"	"1"	"0"
"6e.jpg"	"Ulls"	"Ulls"	"0.39"	"0.61"
"7.jpg"	"NoUlls"	"NoUlls"	"0.99"	"0.01"
"7e.jpg"	"Ulls"	"NoUlls"	"0.67"	"0.33"
"8.jpg"	"NoUlls"	"NoUlls"	"1"	"0"
"8e.jpg"	"Ulls"	"Ulls"	"0.17"	"0.83"
"9.jpg"	"NoUlls"	"NoUlls"	"0.91"	"0.09"
"9e.jpg"	"Ulls"	"Ulls"	"0.32"	"0.68"

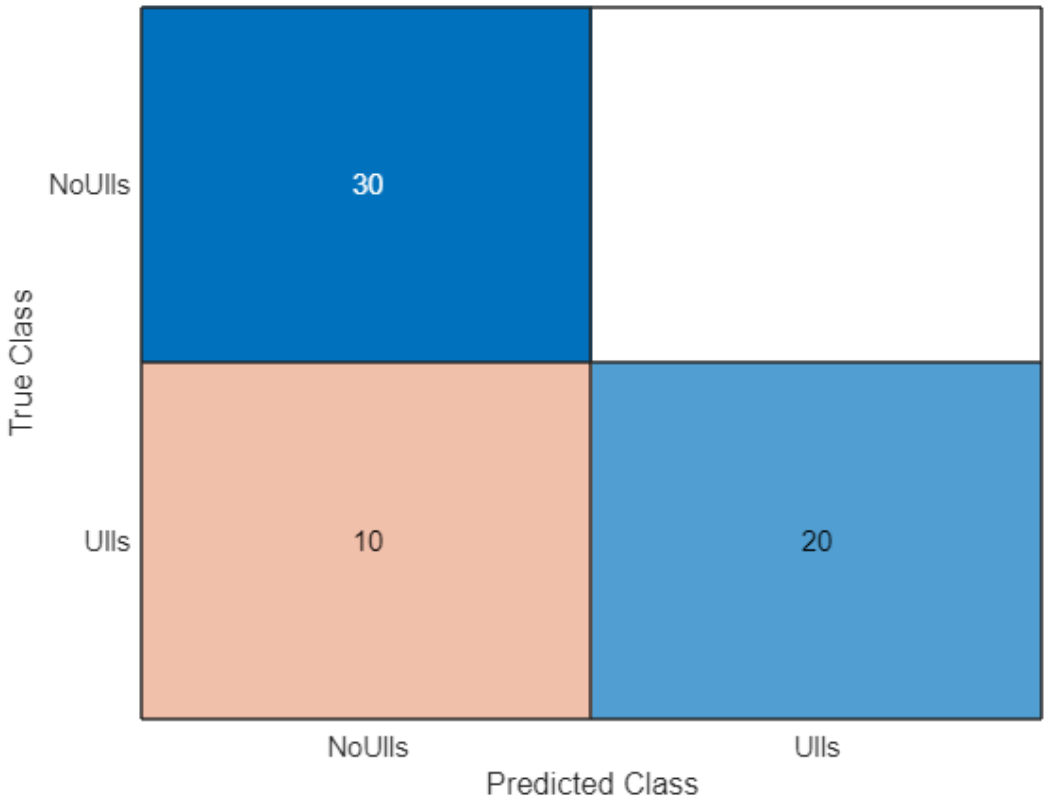

Figura 8: Resultats i Matriu de confusió de TreeBagger amb HOG.

Neural Network

Nom Fitxer	Valor real	Valor predit	Score NoUlls	Score Ulls
"1.jpg"	"NoUlls"	"NoUlls"	"1"	"2.6734e-14"
"1e.jpg"	"Ulls"	"Ulls"	"4.8883e-07"	"1"
"2.jpg"	"NoUlls"	"NoUlls"	"1"	"1.7981e-11"
"2e.jpg"	"Ulls"	"Ulls"	"0.0045674"	"0.99543"
"3.jpg"	"NoUlls"	"NoUlls"	"1"	"1.2269e-14"
"3e.jpg"	"Ulls"	"Ulls"	"2.8397e-07"	"1"
"4.jpg"	"NoUlls"	"NoUlls"	"1"	"7.6843e-14"
"4e.jpg"	"Ulls"	"Ulls"	"6.2962e-07"	"1"
"5.jpg"	"NoUlls"	"NoUlls"	"1"	"5.4116e-16"
"5e.jpg"	"Ulls"	"Ulls"	"4.1261e-06"	"1"
"6.jpg"	"NoUlls"	"NoUlls"	"1"	"1.0063e-13"
"6e.jpg"	"Ulls"	"Ulls"	"1.1018e-06"	"1"
"7.jpg"	"NoUlls"	"NoUlls"	"1"	"8.3004e-15"
"7e.jpg"	"Ulls"	"Ulls"	"6.724e-06"	"0.99999"
"8.jpg"	"NoUlls"	"NoUlls"	"1"	"3.135e-17"
"8e.jpg"	"Ulls"	"Ulls"	"8.2432e-08"	"1"
"9.jpg"	"NoUlls"	"NoUlls"	"1"	"3.1569e-12"
"9e.jpg"	"Ulls"	"Ulls"	"5.0078e-06"	"0.99999"

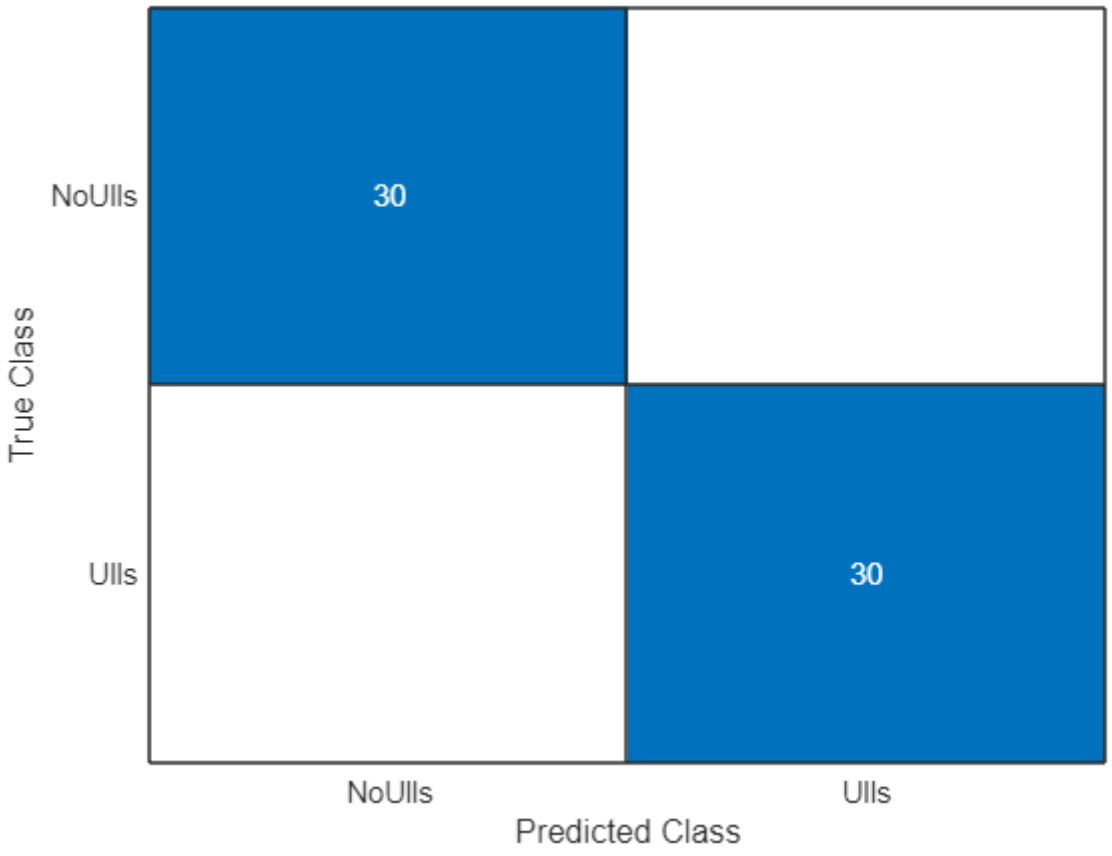

Figura 9: Resultats i Matriu de confusió de TreeBagger amb HOG.

De tots dos classificadors, neural networks obtenia millors resultats. Per a un un conjunt de 60 imatges noves, assolia un 100% d'encerts amb elevada seguretat.

Entrega final

Canvis respecte el Checkpoint

Canvis en l'entrenament del classificador d'ulls

El primer canvi que hem fet ha estat el canvi del vector de característiques, pel Checkpoint vam fer servir un vector de característiques de HOG massa gran (34000 elements), per aquesta entrega vam pensar en fer servir LBPs pero no van donar un bon resultat.

Amb SVM lineal:

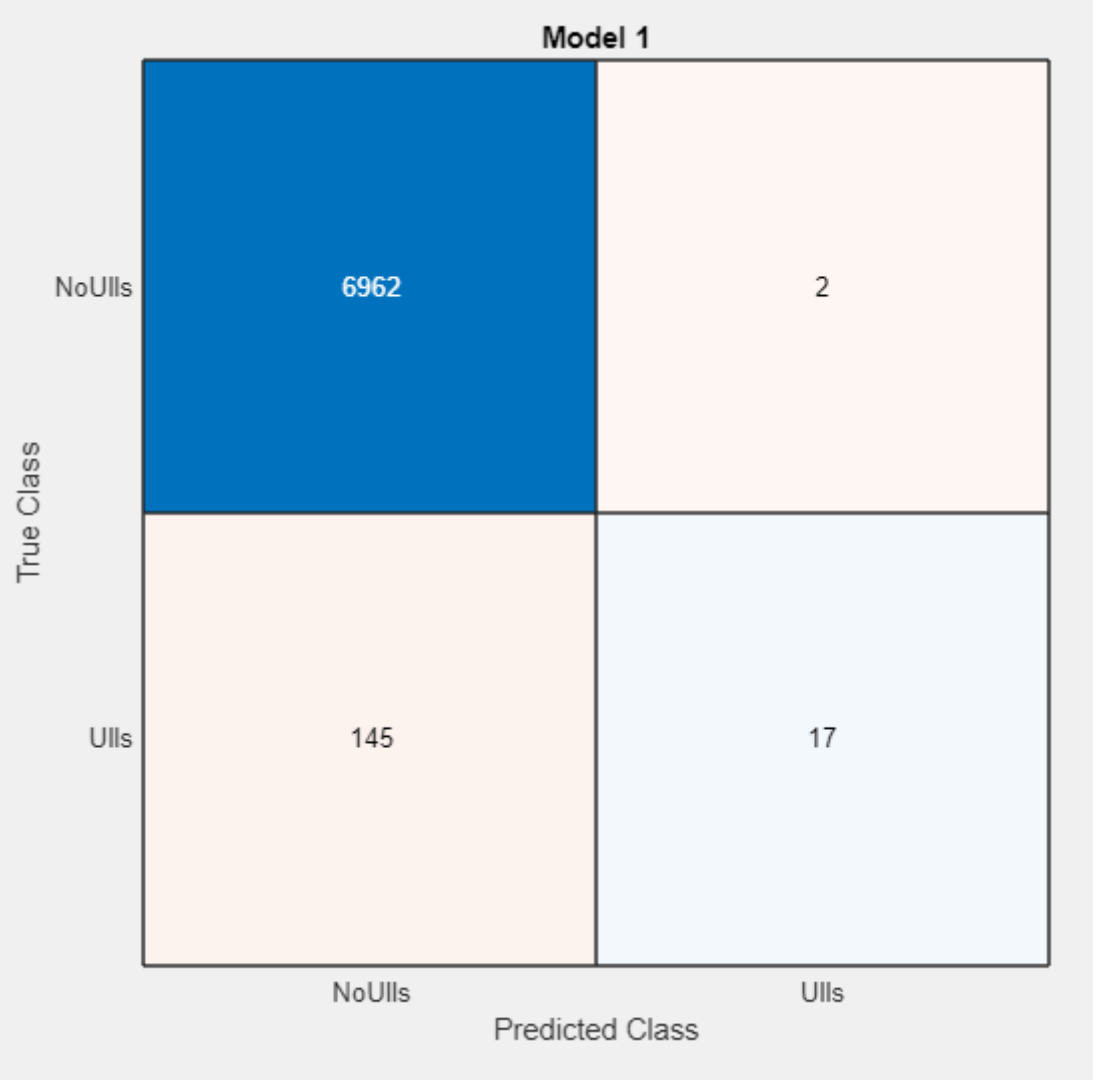

Figura 10: Matriu de confusió de LBPs amb SVM per ulls.

Veiem que molts ulls els classifica com a no ulls.

Amb una xarxa neuronal:

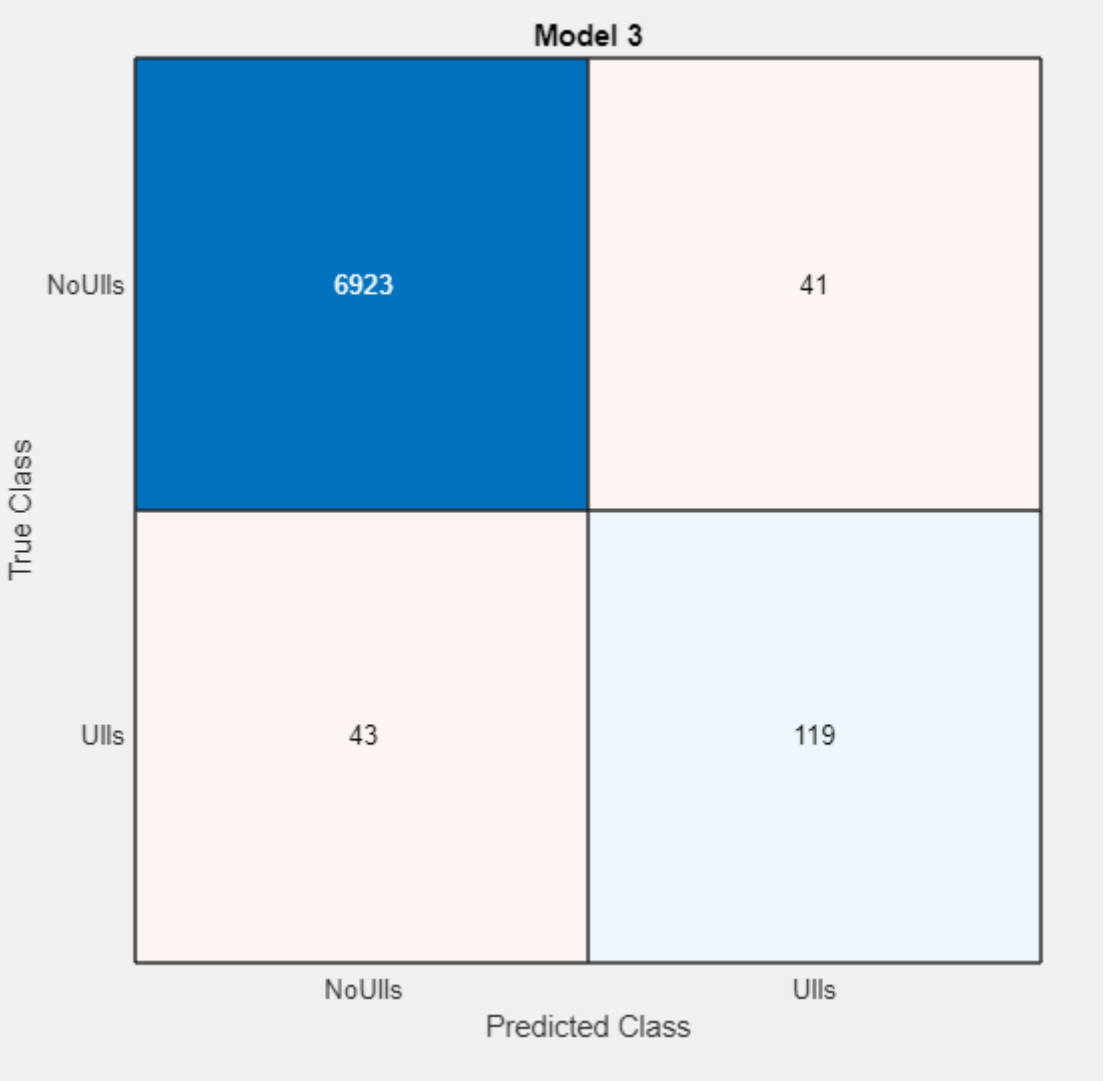

Figura 11: Matriu de confusió de LBPs amb NN per ulls.

Veiem una millora respecte al SVM lineal pero 43 ulls mal classificats son massa.

Al final hem fet servir característiques de HOG amb un cell size de 16x16, això ens deixa un vector de caracterìstiques més petit i més adient per entrenar un classificador.

L'altre canvi que s'ha realitzat respecte al Checkpoint es canviar el classificador de Neural Network a SVM (Support Vector Machines), ja que dona una matriu de confusió prou bona. No hi ha overfitting com amb la xarxa neuronal, i els scores no són estrictement 0 o 1 com ens passava amb NN, sino que van de valors negatius a valors positius. Això permet fer un enquadrament dels ulls més adequat, ja que podem triar la millor de les puntuacions per fer-lo.

Nom Fitxer	Valor real	Valor predit	Score NoUlls	Score Ulls
"1.jpg"	"NoUlls"	"NoUlls"	"1.7361"	"-1.7361"
"10.jpg"	"NoUlls"	"NoUlls"	"1.2909"	"-1.2909"
"10e.jpg"	"Ulls"	"Ulls"	"-0.89405"	"0.89405"
"11.jpg"	"NoUlls"	"NoUlls"	"1.2536"	"-1.2536"
"11e.jpg"	"Ulls"	"Ulls"	"-0.92408"	"0.92408"
"12.jpg"	"NoUlls"	"NoUlls"	"1.4217"	"-1.4217"
"12e.jpg"	"Ulls"	"Ulls"	"-1.1002"	"1.1002"
"13.jpg"	"NoUlls"	"NoUlls"	"1.9999"	"-1.9999"
"13e.jpg"	"Ulls"	"Ulls"	"-1.6653"	"1.6653"
"14.jpg"	"NoUlls"	"NoUlls"	"2.065"	"-2.065"
"14e.jpg"	"Ulls"	"Ulls"	"-1.6716"	"1.6716"
"15.jpg"	"NoUlls"	"NoUlls"	"1.6049"	"-1.6049"
"15e.jpg"	"Ulls"	"Ulls"	"-1.1784"	"1.1784"
"16.jpg"	"NoUlls"	"NoUlls"	"1.5532"	"-1.5532"
"16e.jpg"	"Ulls"	"Ulls"	"-1.7364"	"1.7364"
"17.jpg"	"NoUlls"	"NoUlls"	"1.7247"	"-1.7247"
"17e.jpg"	"Ulls"	"Ulls"	"-1.1552"	"1.1552"
"18.jpg"	"NoUlls"	"NoUlls"	"1.778"	"-1.778"
"18e.jpg"	"Ulls"	"Ulls"	"-1.1392"	"1.1392"
"19.jpg"	"NoUlls"	"NoUlls"	"2.1533"	"-2.1533"
"19e.jpg"	"Ulls"	"Ulls"	"-0.87289"	"0.87289"

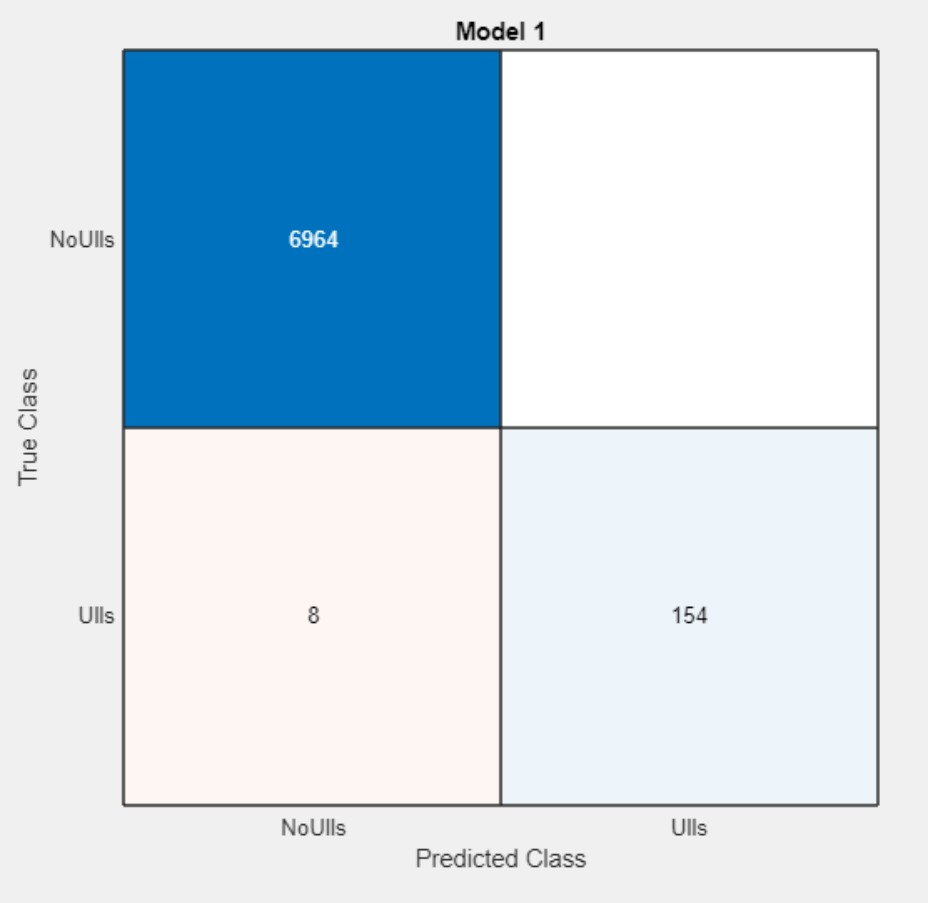

Figura 12: Resultats i Matriu de confusió de HOG amb SVM per ulls.

Base de dades de celles

Per tal d'entrenar el classificador per a la detecció de celles hem emprat el mateix conjunt d'imatges que per a la detecció d'ulls però segmentats de forma diferent. Aquest cop segmentats en imatges de 512x64, mida suficient pel rectangle que conté les celles a la majoria de les imatges. de la mateixa forma que a la base de dades d'ulls en aquest cas també les imatges serán o de celles si corresponen a les celles d'una persona o de no celles en el cas contrari.

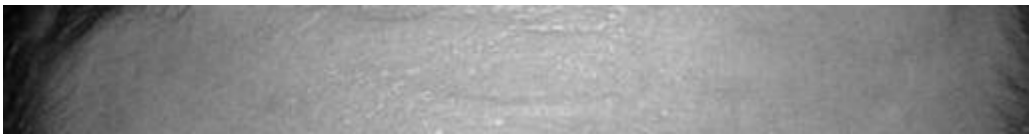

Figura 13: Imatge de característiques diferents a celles.

Figura 14: Imatge de celles.

Classificador de celles

Per a la detecció de celles, després de provar de fer servir LBPs i veure que els resultats no eren adequats, hem fet servir HOG features amb cellsize de [16 16] com amb els ulls.

Hem provat també amb diferents classificadors i ens hem quedat amb el SVM lineal, ja que dona una matriu de confusió prou bona, no hi ha overfitting com amb la xarxa neuronal i els scores no estan són estrictement 0 o 1, sino que es reparteixen entre valors negatius i valors positius. Això ens anirà molt bé per a la detecció de celles en imatges

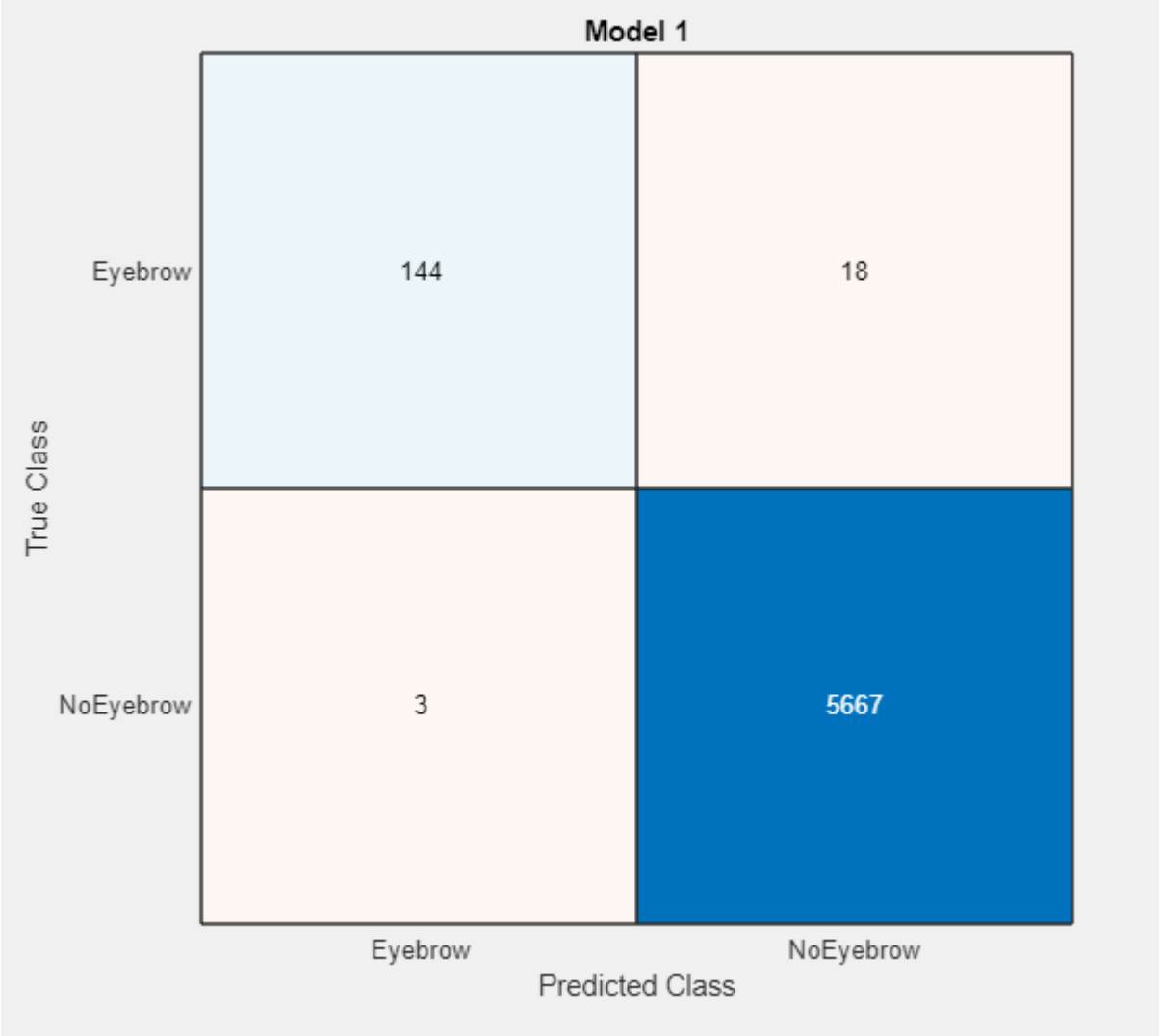

Figura 15: Matriu de confusió de HOG amb SVM per celles.

Podem veure que amb aquest vector de característiques de HOG i fent servir el SVM lineal obtenim un bon resultat que ens permetrà detectar celles correctament.

Detecció de l'iris

Una part fonamental per a detectar la mirada en una foto és localitzar els iris. Per fer-ho, hem optat per trobar cercles en la imatge fent servir la transformada de Hough.

Figura 16: Quadre d'ulls.

Per fer-ho, hem binaritzar la imatge emprant una funció que detecta el llindar adequat per una imatge a escala de grisos, segons el mètode d'*Otsu's threshholding*.

Figura 17: Binaritzat de la imatge.

En aquest pas de binaritzar podem aplicar un open o tècniques similars per tal d'eliminar algunes de les marques i facilitar el procés de detecció, però en general, hem trobat que això no ha estat necessari.

Per tal de detectar tots dos iris, busquem els dos cercles més prominents dins un rang de possibles radis. En la següent imatge els marquem amb un punt vermell:

Figura 18: Centre dels iris marcats.

Tot i ser un procediment que depèn que el iris sigui visible parcialment (una imatge amb els ulls molt tancats no serviría), amb les mostres d'ulls que tenim de la base de dades i d'altres de fora de la base de dades ha donat bons resultats.

Detecció del llagrimal i de la comissura

Una altre part necessària per tal de detectar la mirada és identificar els extrems de l'ull. Per fer-ho, segmentem verticalment la imatge per a tractar els ulls per separat. Ho podem fer amb facilitat, ja que partim d'un conjunt de fotos on les cares estan més o menys centrades. Per tal de detectar els extrems de l'ull emprem les característiques de Harris.

Figura 19: Característiques de Harris.

Aquí separem el procés en dos: la detecció del llagrimal i la detecció de la comissura.

Per tal de detectar el llagrimal, fem un binaritzat inicial de la imatge. D'aquesta, en moltes ocasions serà necessari fer un close per eliminar taques negres, ja que la frontera del nas pot interferir amb l'elecció del llagrimal. Per trobar el llagrimal, el que fem és usar les característiques de Harris amb un nivell de qualitat baix per tal de trobar molts punts en una regió fitada entre el centre de l'ull i el nas. De tots aquests ens quedem amb el punt més a la dreta (en el cas d'estar tractant l'ull dret de la persona). Ja que la qualitat mínima dels detalls és baixa, és necessari fer el close per tal que el nas, com s'ha dit, no interfereixi al seleccionar el punt desitjat.

Per altra banda, la detecció de la comissura resulta més senzilla, ja que la petita secció cabell presentarà menys interferència. Aquí ens podem saltar el pas de close, i quedar-nos tan sols en l'extrem oposat de les característiques de Harris obtingudes entre el centre de l'ull i el cabell.

Amb aquests dos punts tenim marcats els dos extrems possibles de l'ull.

Detecció de mirada

Per detectar la mirada hem fet servir la detecció de l'iris i la detecció del llagrimal i cantonada dels ulls, explicades anteriorment. Amb la informació que aquestes ens donen, podem deduir el desplaçament horitzontal de l'iris respecte del llagrimal.

Figura 20: Quadre d'ulls a processar.

Considerem que amb detectar un dels dos iris n'hi ha prou per tal de definir la mirada. Així doncs, repetim el procés de detecció de l'iris i del llagrimal:

Figura 21: Binaritzat de l'imatge.

Amb la posició del centre de l'iris podem afinar el procés d'obtenció del llagrimal, prenent les característiques de Harris més properes a aquest.

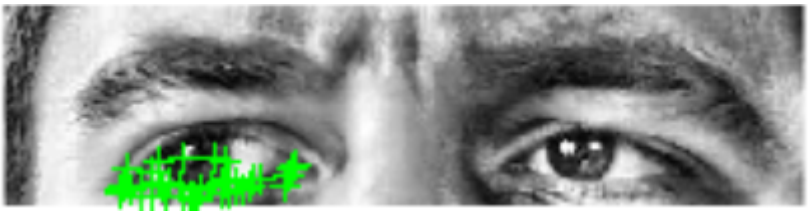

Figura 22: Característiques de Harris de l'ull.

Havent obtingut les coordenades de tant l'iris, el llagrimal, i la cantonada de l'ull, podem mesurar la distància horitzontal entre el centre de l'iris i l'ull, i deduir informació sobre en quina direcció està mirant l'individu.

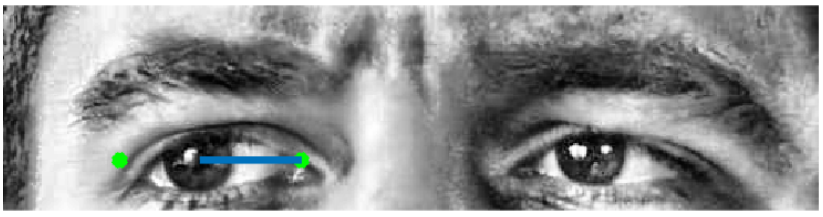

Figura 23: Posicions horitzontals de l'iris, llagrimal, comissura, i distància horitzontal entre l'iris i el llagrimal.

Detecció i distància entre celles

La següent tasca a realitzar és la detecció de celles. Per a fer-ho, segmentem la imatge verticalment per tal de separar la cella dreta de la cella esquerra (ja que partim de la suposició que la cara estarà més o menys centrada en la imatge). Binaritzem cada una d'aquestes i li apliquem un close morfològic per treure petites taques negres que puguin interferir amb la detecció dels extrems de les celles.

.

Figura 24: Binaritzat d'una cella.

Un cop fet això, detectem els contorns amb la funció detectHarrisFeatures i ens quedem amb la característica de component horitzontal màxima (és a dir, l'extrem de la dreta) per la cella esquerra, i la característica de component horitzontal mínima de la segona meitat (és a dir, l'extrem de la esquerre) de la cella de la dreta.

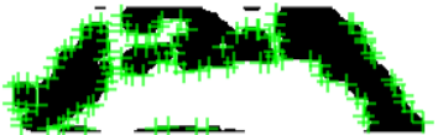

Figura 25: Característiques de Harris d'una cella.

Amb aquestes dues components podem calcular i representar la distancia de l'entrecella.

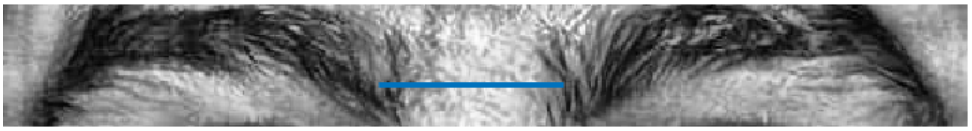

Figura 26: Distància entre celles

Detecció en vídeos

Per fer la detecció de les faccions en vídeos, es separa el vídeo en els seus frames individuals i es segueix un procés similar al de tractament de fotos.

Primer de tot, a diferència de en la nostra base de dades, les cares en un video generalment no estan centrades en el fotograma. Per fer això usem *CascadeObjectDetector*, amb la que retallarem el quadre corresponent a la cara, i aplicarem tot el procés anterior de detecció de característiques. Simplement es segmentarien les regions desitjades, es calcularien els quadres i punts d'interès, i se'ls aplicaria un escalat i translació abans de pintar-los sobre cada un dels fotogrames originals, obtenint el vídeo final.

Aquest procediment ha resultat força imprecís, doncs amb els vídeos provats no es detectava la cara amb eficàcia (de vegades veia cares on no n'hi havien, i a l'inrevés), i de les que sí que detectava, a causa de diversos motius com la il·luminació, la qualitat, l'angle, i les pròpies mancances del nostre algoritme, tenia problemes per detectar-ne els trets.

A causa de l'elevat cost de tractar amb vídeos, aquestes mancances no s'han placat prou, i el codi actual encara no és del tot capaç d'enfrontar-se amb vídeos de manera efectiva i precisa.

Resultats i conclusions

Pel que fa a la detecció de totes les característiques (ulls, celles, entre-cella i mirada) en imatges de cares, els resultats son molt bons per a cares similars a les que hem fet servir per entrenar els classificadors (imatges extretes de *thispersondoesnotexist.com*). Els resultats per imatges amb diferent resolució i que no han sigut extretes d'aquesta pàgina també és força bo. Generalment, amb la resolució i jugant amb alguns paràmetres, el programa és capaç de detectar tots els trets desitjats amb precisió.

Degut a l'ús de tècniques com la binarització o trobar les característiques de Harris, que no són invariants als canvis de lluminositat, hem afegit unes variables ajustables per que el resultat sigui correcte. Cal dir que amb les variables per defecte (sense canviar els valors) el programa troba les característiques de gran part de les imatges correctament, però en alguns casos sí que resulta necessari ajustar-les.

Aquestes variables són les següents:

- alpha és un paràmetre que ajusta el llindar de binaritzat. En alguns casos cal incrementar-la si el llindar d'Otsu no es adequat. Pren 0 com a valor per defecte.
- minQllag és un paràmetre que indica la qualitat mínima de les cantonades acceptades del costat del llagrimal. En alguns casos cal incrementar-la o reduir-la per permetre acceptar més punts o menys. Pren 0.01 com a valor per defecte.
- minQcontllag és un paràmetre que indica la qualitat mínima de les cantonades acceptades del costat contrari al del llagrimal. En alguns casos cal incrementar-la o reduir-la per permetre acceptar més punts o menys. Pren 0.01 com a valor per defecte.

Per demostrar el correcte funcionament del programa, s'ha fet la detecció de característiques per les següents imatges provinents de diferents llocs amb les els valors per defecte de les variables.

Exemples extrets de *thispersondoesnotexist.com*:

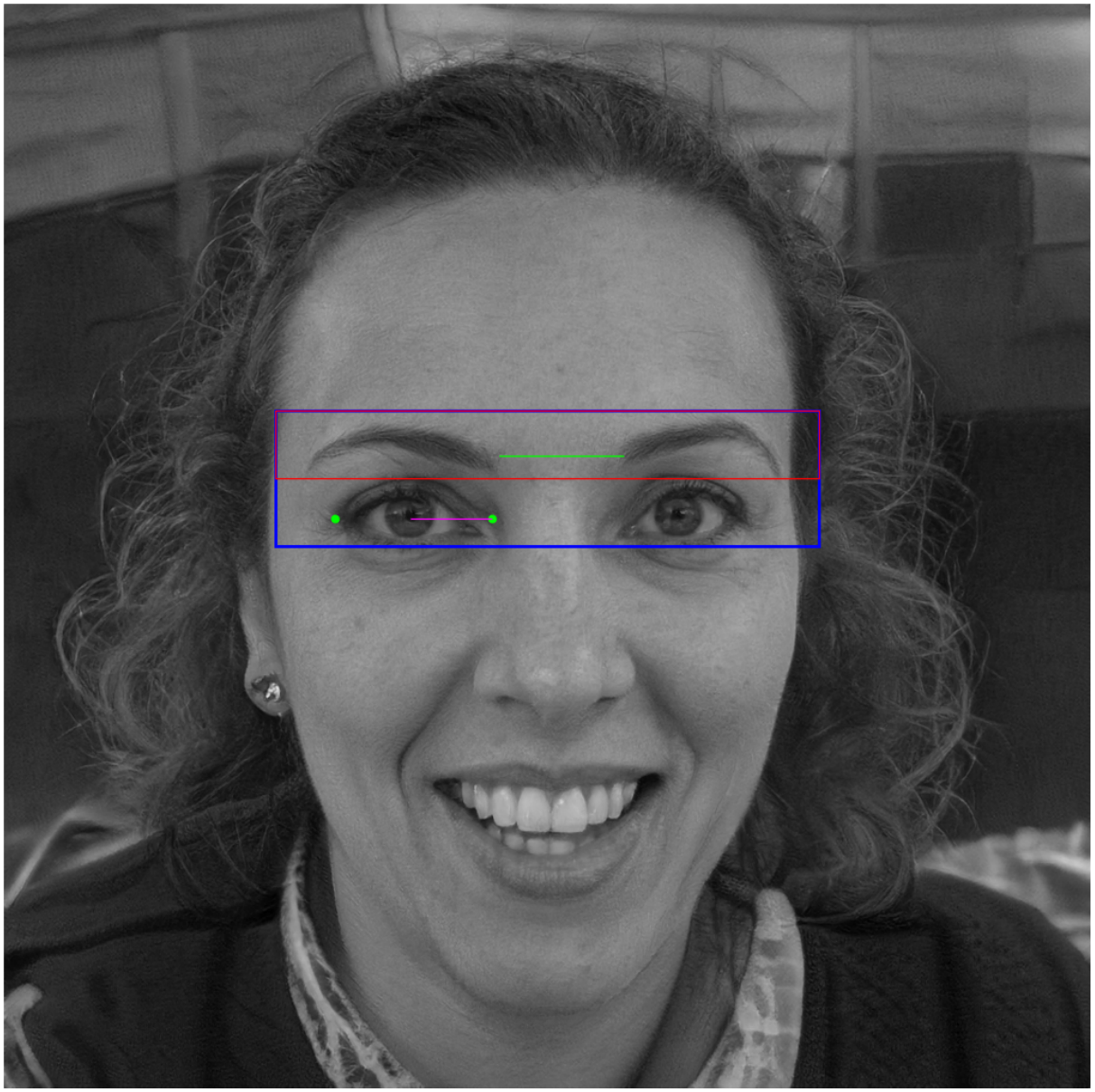

Figura 27: Trets d'una dona generada per IA.

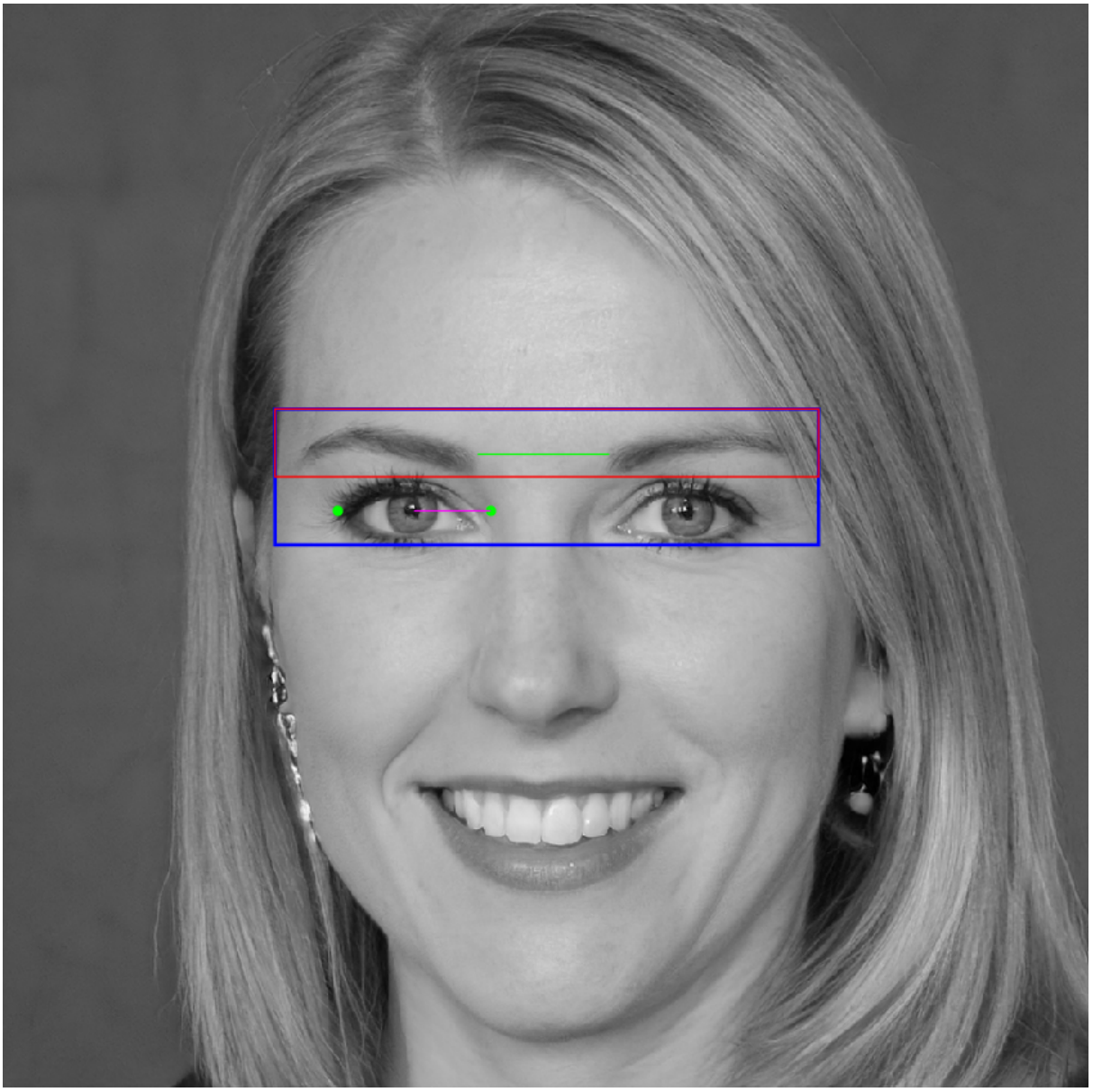

Figura 28: Trets d’una dona generada per IA. (2)

Imatges d’humans reals:

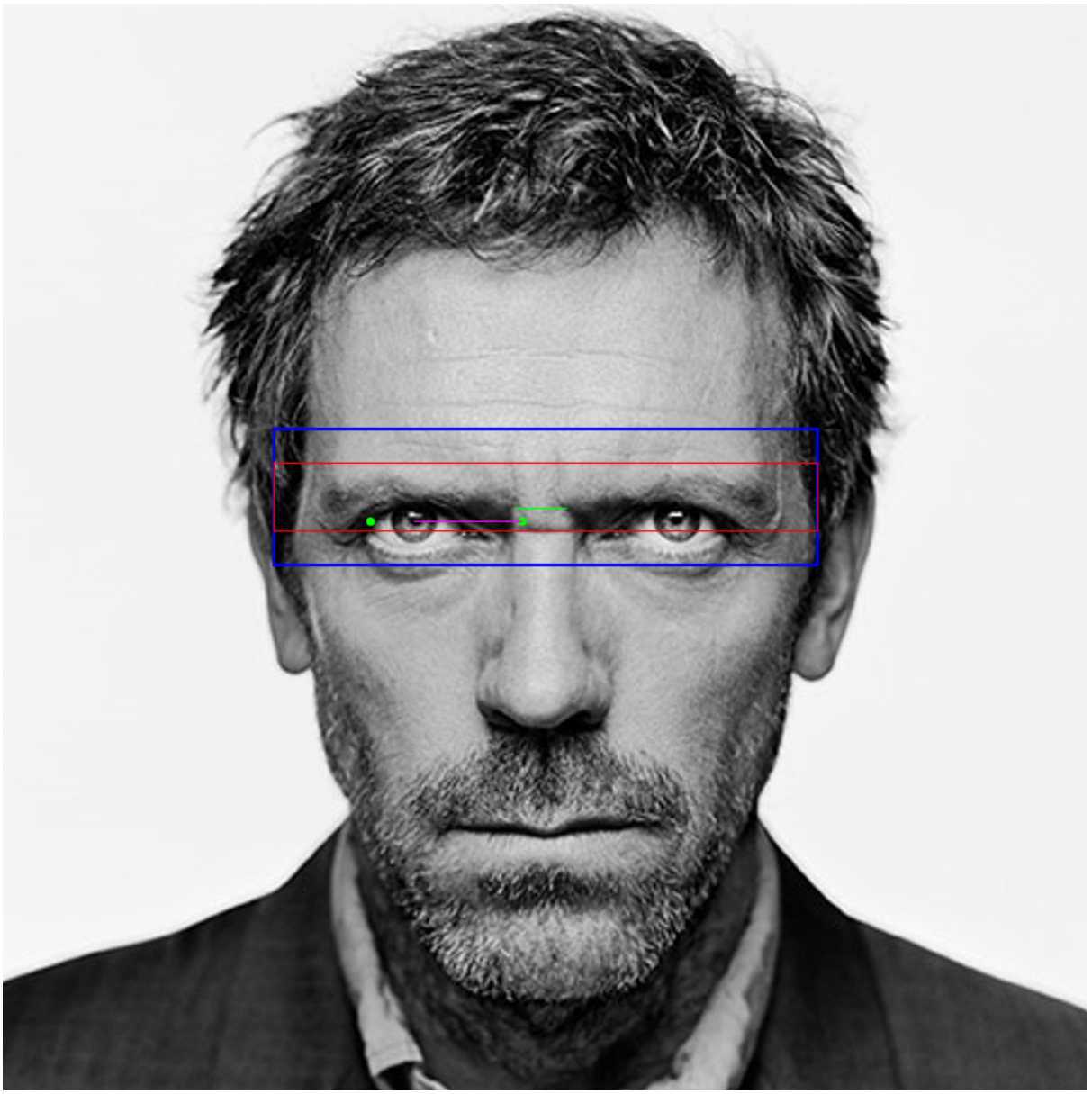

Figura 29: Detecció dels trets de Hugh Laurie.

En general a trobat bé tots els trets, a excepció del llagrimal que, degut al plec del nas, hi ha hagut una mica d’interferència en la seva detecció, i ha quedat desplaçat cap a la dreta.

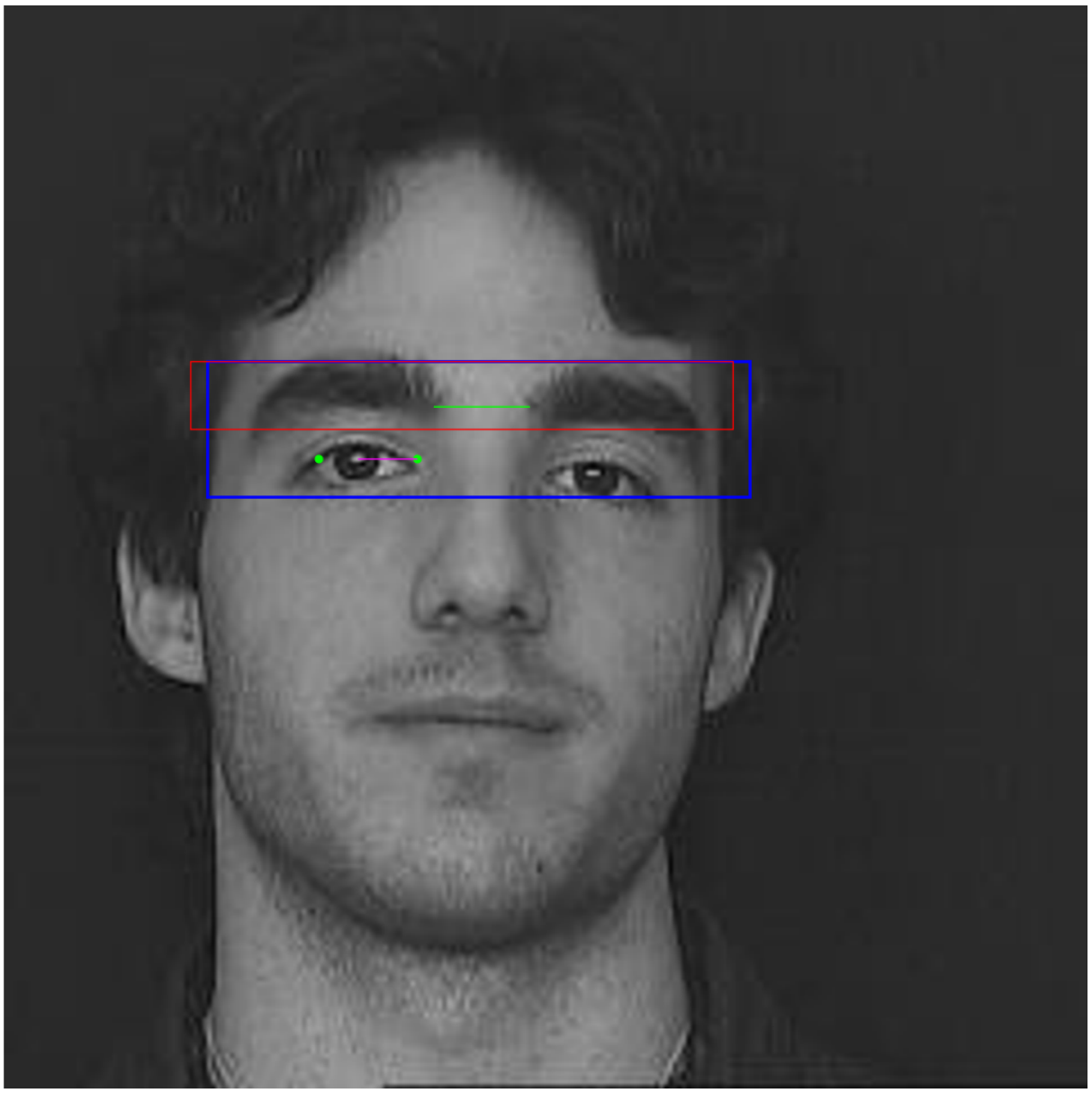

Figura 30: Detecció dels trets d'un home

En aquest cas troba els trets amb força precisió.

Annex:

Totes les funcions han estat implementades per Francesc Forn, Raúl Lumbreras i Martí Vall

Codi per fer la base de dades de Ulls:

```
files = dir('images_noulleres\*.jpg');
N = length(files);


for n = 1:N
    I = rgb2gray(imread(files(n).name));
    s = strcat('eyesImages\',int2str(n),'.jpg');
    imwrite(I((256+128):(256+256-1), 256:(256+512-1)),s)
end
```

Codi per fer la base de dades de NoUlls:

```
files = dir('images\*.jpg');
N = length(files);
list = [1 128 256 384 512 640 768 896]


for n = 1:N
    I = rgb2gray(imread(files(n).name));
    for i = 1:length(list)
        if list(i) ~= 384
            for j = 1:length(list)-3
                s = strcat('noeyesImages\',int2str(n),'_',int2str(i),'_',int2str(j),'.jpg');
                imwrite(I(list(i):list(i)+127, list(j):list(j)+511),s)
            end
        end
    end
end
```

Codi per fer la base de dades de Eyebrows:

```
files = dir('C:\Users\Raúl\Documents\MATLAB\VC\Projecte\images_noulleres\*.jpg');
N = length(files);


for n = 1:N
    I = rgb2gray(imread(files(n).name));
    s = strcat('C:\Users\Raúl\Documents\MATLAB\VC\Projecte\eyebrowsImages\',int2str(n),'.jpg');
    imwrite(I((256+128):(256+128+64-1), 256:(256+512-1)),s)
end
```

Codi per fer la base de dades de NoEyebrows:

```
files = dir('C:\Users\Raúl\Documents\MATLAB\VC\Projecte\images_noulleres\*.jpg');
N = length(files);
list = [1 128 256 384 512 640 768 896]


for n = 1:N
    I = rgb2gray(imread(files(n).name));
    for i = 1:length(list)
        if list(i) ~= 384
            for j = 1:length(list)-3
                s = strcat('C:\Users\Raúl\Documents\MATLAB\VC\Projecte\noeyebrowsImages\',int2s
                imwrite(I(list(i):list(i)+63, list(j):list(j)+511),s)
            end
        end
    end
end
```

Codi de feature vectors amb histogrames de grisos:

```
files = dir('Projecte\eyesImagesRect\*.jpg');
desc = []
out = {}
for n = 1:length(files)
    I = imread(files(n).name);
    h = histogram(I,30);
    desc = [desc; h.Values];
    out = [out, "Ulls"];
end
files = dir('Projecte\noeyesImagesRect\*.jpg');
for n = 1:length(files)
    I = imread(files(n).name);
    h = histogram(I,30);
    desc = [desc; h.Values];
    out = [out, "NoUlls"];
end
```

Codi de generació de les imatges de test d'ulls:

```
files = dir('Projecte\test\*.jpg');
N = length(files);


for n = 1:N
    I = rgb2gray(imread(files(n).name));
    Ulls = strcat(int2str(n+1),'e.jpg');
    imwrite(I((256+128):(256+256-1), 256:(256+512-1)),Ulls)
    NoUlls = strcat(int2str(n+1),'.jpg');
    imwrite(I((256+256+128):(256+256+256-1), 256:(256+512-1)),NoUlls)
```

```
end
```

Codi de generació de les imatges de test de celles:

```
files = dir('Projecte\testEyebrows\*.jpg');
N = length(files);


for n = 1:N
    I = rgb2gray(imread(files(n).name));
    Brow = strcat(int2str(n+1),'b.jpg');
    imwrite(I((256+128):(256+128+64-1), 256:(256+512-1)),Brow)
    NoBrow = strcat(int2str(n+1),'.jpg');
    imwrite(I((256+256+128):(256+256+128+64-1), 256:(256+512-1)),NoBrow)

end
```

Codi de feature vectors amb HOGs per ulls:

```
files = dir('Projecte\eyesImagesRect\*.jpg');
desc = [];
out = {};
for n = 1:length(files)
    I = imread(files(n).name);
    %featuresF = extractLBPFeatures(I);
    featuresF = extractHOGFeatures(I,CellSize = [16 16]);
    desc = [desc; featuresF];
    out = [out, "Ulls"];
end
files2 = dir('Projecte\noeyesImagesRect\*.jpg');
for n = 1:length(files2)
    I = imread(files2(n).name);
    %featuresF = extractLBPFeatures(I);
    featuresF = extractHOGFeatures(I,CellSize = [16 16]);
    desc = [desc; featuresF];
    out = [out, "NoUlls"];
end

%Classificador = TreeBagger(100,desc,out');
%Classificador  = fitcnet(desc,out','Standardize',true)
Classificador = fitcsvm(desc,out');
```

Predició del test i creació de la taula per ulls:

```
files = dir('Projecte\test\*.jpg');
N = length(files);
res = [];
for n = 1:N
    I = imread("test\" + files(n).name);
```

```
    files(n).name;
    features = extractHOGFeatures(I);
    %descTest = [descTest; features];
    k = strfind(files(n).name,'e');
    if k > 0
        outt = "Ulls";
    else
        outt = "NoUlls";
    end
    [label,score] = predict(Classifier,[features]);
    res = [res; files(n).name, outt, label, score];
end

array2table(res,"VariableNames",["Nom Fitxer" "Valor real" "Valor predit" "Score NoUlls" "Score
```

Codi de feature vectors amb HOGs per celles:

```
files = dir('Projecte\eyebrowsImages\*.jpg');
desc = [];
out = {};
for n = 1:length(files)
    I = imread(files(n).name);
    featuresF = extractHOGFeatures(I,CellSize= [16 16]);
    desc = [desc; featuresF];
    out = [out, "Eyebrow"];
end
files2 = dir('Projecte\noeyebrowsImages\*.jpg');
for n = 1:(length(files2))
    I = imread(files2(n).name);
    featuresF = extractHOGFeatures(I,CellSize= [16 16]);
    desc = [desc; featuresF];
    out = [out, "NoEyebrow"];
end

%Classificador = TreeBagger(50,desc,out');
%Classificador  = fitcnet(desc,out','Standardize',true)
Classificador = fitcsvm(desc,out);
```

Predició del test i creació de la taula per celles:

```
files = dir('Projecte\testEyebrows\*.jpg');
N = length(files);
res = [];
for n = 1:N
    I = imread("Projecte\testEyebrows\" + files(n).name);
    files(n).name;
    featuresF = extractHOGFeatures(I,CellSize= [16 16]);
    %descTest = [descTest; features];
    k = strfind(files(n).name,'b');
    if k > 0
```

```
        outt = "Eyebrow";
    else
        outt = "NoEyebrow";
    end
    [label,score] = predict(Classificador,[featuresF]);
    res = [res; files(n).name, outt, label, score];
end
res
array2table(res,"VariableNames",["Nom Fitxer" "Valor real" "Valor predit" "Score NoUlls" "Score
```

Detecció d'iris en imatges d'ulls:

```
I = imread("eyesImagesRect\3.jpg");
imshow(I);

Ibin = I < 120
imshow(Ibin)
SE = strel("disk", 3)
Ibin = imerode(Ibin, SE)
imshow(Ibin)

centers = round(imfindcircles(Ibin,[16 48]));

I(:,:,2) = I(:,:,1);
I(:,:,3) = I(:,:,1);
I(centers(1,2):centers(1,2)+5,centers(1,1):centers(1,1)+5,1) = 255
I(centers(2,2):centers(2,2)+5,centers(2,1):centers(2,1)+5,1) = 255

imshow(I);
```

Detecció de llagrimal i el seu contrari en imatges d'ulls:

```
I = adapthisteq(imread("eyesImagesRect\3.jpg"));
imshow(I);

Ibin = I <120

SE = strel("disk", 3);

Ibin = imclose(Ibin, SE)
imshow(Ibin)

Ibin = Ibin(:,1:255)
centers = round(imfindcircles(Ibin,[16 48], Sensitivity=0.95));

fllag = detectHarrisFeatures(I,ROI = [round(centers(1,1)), round(centers(1,2)), 75, size(I,1)-r
fcontllag = detectHarrisFeatures(I,ROI = [round(centers(1,1))-75, round(centers(1,2)), 75, size
imshow(I)
hold on
```

```
plot(fllag.selectStrongest(20))

cPllag = fllag.selectStrongest(20)

plot(fcontllag.selectStrongest(20))
hold off
cPcontllag = fcontllag.selectStrongest(20)

xmax =round(max(cPllag.Location(:,1)))
xmin =round(min(cPcontllag.Location(:,1)))


xmax = round(xmax);
I(:,:,2) = I(:,:,1);
I(:,:,3) = I(:,:,1);
%I(centers(1,2):centers(1,2)+5,centers(1,1):centers(1,1)+5,1) = 255
%I(centers(1,2):centers(1,2)+5, xmax:xmax+5, 3) = 255
%I(centers(1,2):centers(1,2)+5, xmin-5:xmin, 3) = 255


imshow(I);
hold on
plot(xmax,centers(1,2),"g*", LineWidth = 10)
plot(xmin,centers(1,2),"g*", LineWidth = 10)
line([centers(1,1) xmax], [centers(1,2) centers(1,2)], LineWidth = 5 )
hold off
```

Calcul de l'entrecella en imatges de celles:

```
I = adapthisteq(imread("eyebrowsImages\1.jpg"));

Iesq = I(:,1:size(I,2)/2);
Idre = I(:,size(I,2)/2 +1:end);

thEsq = graythresh(Iesq);
Iesqbin = imbinarize(Iesq,thEsq);
SE = strel("disk", 5);
Iesqbin = imclose(Iesqbin,SE);
imshow(Iesqbin);

puntsEsq = detectHarrisFeatures(Iesqbin, MinQuality= 0.1)

xEsq = max(puntsEsq.Location(:,1))

thDre = graythresh(Idre);
Idrebin = imbinarize(Idre,thDre);

Idrebin = imclose(Idrebin,SE);
imshow(Idrebin);
hold on
puntsDre = detectHarrisFeatures(Idrebin, MinQuality= 0.1)
```

```
xDre = min(puntsDre.Location(:,1))

plot(puntsDre);
hold off

imshow(I);
hold on
line([xEsq (size(I,2)/2)+xDre], [size(I,1)*2/3 size(I,1)*2/3], LineWidth = 3 )
hold off
```

Detecció de tot en imatges de cares:

```
load ClassificadorEyebrows.mat;
ClassEyebrows = Classificador
load Classificador.mat;

I = imresize(rgb2gray(imread("Projecte\testImg.jpg")),[1024 1024])
imshow(I)
hold on
[n,m] = size(I) %1024 1024 -> 512 128

b = 64;
maxScore = 0;
maxScore2 = -3;
maxi = 0;
maxj = 0;
maxiE = 0;
maxjE = 0;

for j = 1:b:(m-512)

    for i = 1:b:(n-64)
        Im = I(i:(i+64-1), j:(j+512-1));
        features = extractHOGFeatures(Im, CellSize= [16, 16]);
        [label, score] = predict(ClassEyebrows, features);
        if(score(1) > maxScore)
            maxScore = score(1);
            maxiE = i
            maxjE = j
        end
    end

    for i = 1:b:(n-128)
        Im = I(i:(i+128-1), j:(j+512-1));
        features = extractHOGFeatures(Im, CellSize= [16, 16]);
        [label, score] = predict(Classificador, features);
        if(score(2) > maxScore2)
            maxScore2 = score(2)
            maxi = i;
            maxj = j;
        end
    end
```

```
end

rectangle('Position',[maxj,maxi, 512,128],'EdgeColor','b','LineWidth',2)
rectangle('Position',[maxjE,maxiE, 512,64],'EdgeColor','r','LineWidth',1)

alpha = 0
minQllag = 0.01
minQcontllag = 0.01

Irec = I(maxiE:maxiE+64-1,maxjE:maxjE+512-1);
Iesq = I(maxiE:maxiE+64-1,maxjE:maxjE+256-1);
Idre = I(maxiE:maxiE+64-1,maxjE+256:maxjE+512-1);

thEsq = graythresh(Iesq);
Iesqbin = imbinarize(Iesq,thEsq);
SE = strel("disk", 5);
Iesqbin = imclose(Iesqbin,SE);

puntsEsq = detectHarrisFeatures(Iesqbin, MinQuality= 0.1)

xEsq = max(puntsEsq.Location(:,1))

thDre = graythresh(Idre);
Idrebin = imbinarize(Idre,thDre);

Idrebin = imclose(Idrebin,SE);

puntsDre = detectHarrisFeatures(Idrebin, MinQuality= 0.1)

xDre = min(puntsDre.Location(:,1))

line([maxjE+xEsq maxjE+(size(Irec,2)/2)+xDre], [maxiE+size(Irec,1)*2/3 maxiE+size(Irec,1)*2/3],

IrecEye = I(maxi:maxi+128-1,maxj:maxj+256-1);
IrecEye2 = adapthisteq(IrecEye);

th = graythresh(I);
Ibin = 1-imbinarize(IrecEye2,th+alpha);

Ibin = Ibin(:,1:255)

[centers,radis] = imfindcircles(Ibin,[16 48], Sensitivity=0.96);
centers = round(centers);
SE = strel("disk", 3);
Ibin = imopen(Ibin, SE)

fllag = detectHarrisFeatures(Ibin,ROI = [round(centers(1,1)), round(centers(1,2)), size(IrecEye
fcontllag = detectHarrisFeatures(IrecEye,ROI = [1, round(centers(1,2)), round(centers(1,1)), si


cPllag = fllag.selectStrongest(20)
```

```
cPcontllag = fcontllag.selectStrongest(20)

xmax =round(max(cPllag.Location(:,1)))
xmin =round(min(cPcontllag.Location(:,1)))

xmax = round(xmax);
IrecEye(:,:,2) = IrecEye(:,:,1);
IrecEye(:,:,3) = IrecEye(:,:,1);

plot(maxj+xmax,maxi +centers(1,2),"g*", LineWidth = 2)
plot(maxj+xmin,maxi +centers(1,2),"g*", LineWidth = 2)
line([maxj + centers(1,1) maxj + xmax], [maxi + centers(1,2) maxi + centers(1,2)], LineWidth =
hold off
```

Detecció d'ulls en videos amb cares:

```
%Create a cascade detector object.
Detector = vision.CascadeObjectDetector('FrontalFaceLBP');

% Read a video frame and run the face detector.
videoReader = VideoReader('videocara.mp4');

video = VideoWriter('facevideo');
open(video);

while hasFrame(videoReader)
    % get the next frame
    videoFrame = readFrame(videoReader);
    bbox = step(Detector, videoFrame);
    if size(bbox,1) ~= 0
        for elem = 1:size(bbox,1)
        I = imresize(videoFrame(bbox(1,2):bbox(1,2)+bbox(1,4) ,bbox(1,1):bbox(1,1)+bbox(1,3)),
        [n,m] = size(I); %1024 1024 -> 512 128

        b = 64;
        maxScore = 0;
        maxi = 0;
        maxj = 0;
        for j = 1:b:(n-128)
            for i = 1:b:(m-512)
                Im = I(j:(j+128-1), i:(i+512-1));
                features = extractHOGFeatures(Im, CellSize= [16 16]);
                [label, score] = predict(Classificador, features);
                if(score(2) > 0.5 && score(2) >= maxScore)
                    maxScore = score(2)
                    maxi = i;
                    maxj = j;
                end
            end
        end
        %rectangleUlls = [i,j, i+512-1,j+128-1]
        maxScore;
```

```
        rectangle('Position',[maxi,maxj, 512,128],'EdgeColor','r','LineWidth',2)
        x0 = bbox(1);
        y0 = bbox(2);
        bboxres = [x0 + maxi*bbox(3)/1024,y0 + maxj*bbox(4)/1024, 512*bbox(3)/1024,128*bbox(4)/
        videoFrame = insertShape(videoFrame, 'Rectangle', bboxres, Color = 'red');
        end
    end
    % Draw the returned bounding box around the detected face.
    videoFrame = insertShape(videoFrame, 'Rectangle', bbox);
    writeVideo(video,videoFrame);
end
close(video);
```