Enginyeria en Informàtica

Projecte de final de Carrera
Universitat Oberta de Catalunya

Identificació de subjectes a partir de l'anàlisi de dades multimodals RGB-Depth

Visió per computador

Autor: Joan Herman i Balaguer
Directors: Sergio Escalera Guerrero
Albert Clapès i Sintes

Barcelona, desembre de 2012

Agraïments

En primer lloc vull expressar el meu agraïment als directors d'aquest projecte: al Sergio pel seu ajut a l'hora de reorientar–lo i pels seus comentaris i correccions per dur-lo a bon port; i a l'Albert amb el que sempre he pogut comptar per ajudar-me amb l'entorn de treball i els problemes amb el codi.

També als estudiants de Máster i Doctorat del Departament de Matemàtica Aplicada i Anàlisi de la Universitat de Barcelona per la seva col·laboració i en especial al Victor Ponce pel seu ajut sempre que l'he necessitat.

A la Isabel, per seu suport i paciència que han fet possible l'arribar a aquesta fita i tot allò que significa.

Al Ramon per creure que ho aconseguiria quan ni tan sols hi havia començat.

I als meus fills el Jan i la Sara, per totes les estones que no els he pogut dedicar.

GRÀCIES

Resum

A la societat actual, connectada gairebé a nivell global, existeix una necessitat creixent de determinar o verificar la identitat de les persones. Aquesta necessitat es fa patent en molts aspectes de la vida quotidiana on es requereix una autorització per l'accés: des del control de presència en una companyia, fins a l'accés a l'embarcament d'una aeronau.

Malgrat els darrers avenços tecnològics, la identificació i autentificació de persones esdevé un repte important quan aquesta s'ha de fer de manera automàtica amb precisió i no repudi, ja que els mètodes actuals basats en contrasenyes i targetes identificatives amb fotografia esdevenen ineficaços quan se'ls posa davant de la forta demanda actual en alguns entorns concrets.

En aquest treball s'explora el camp de la identificació facial de subjectes utilitzant tècniques d'anàlisi multimodal, això és utilitzant imatges RGB i imatges de profunditat (3D) amb l'objecte de validar les diverses tècniques emprades en el reconeixement facial i aprofundir en sistemes que incorporen informació tridimensional als algorismes de detecció i identificació facial.

Per últim s'ha dissenyat un prototipus que utilitza la comparació de punts de característiques facials entre dos subjectes i tècniques d'Homografia, per tal de millorar els conjunts d'entrenament d'un algorisme d'anàlisi de components principals (PCA), i així obtenir uns resultats més robustos i amb un índex de fiabilitat elevat respecte a un conjunt d'entrenament estàndard.

Resumen

En la sociedad actual, prácticamente conectada a nivel global, existe una necesidad creciente de determinar o verificar la identidad de las personas. Esta necesidad se hace patente en muchos aspectos de la vida cotidiana donde es requiere una autorización para el acceso: desde el control de presencia en una compañía, hasta el acceso al embarque de una aeronave.

A pesar de los últimos adelantos tecnológicos, la identificación y autentificación de personas acontece un reto importante cuando ésta se tiene que hacer de manera automática con precisión y no repudio. Esto es debido a que los métodos actuales basados en contraseñas y tarjetas identificativas con fotografía resultan ineficaces cuando se los somete a la fuerte demanda actual de algunos entornos concretos.

En este trabajo se explora el campo de la identificación facial de sujetos utilizando técnicas de análisis multimodal, esto es utilizando imágenes RGB e imágenes de profundidad (3D), con el objeto de repasar las diversas técnicas empleadas en el reconocimiento facial y profundizar en sistemas que incorporan información tridimensional a los algoritmos de detección e identificación facial.

Por último se ha diseñado un prototipo que utiliza la comparación de puntos de

características faciales entre dos sujetos y técnicas de homografía, para mejorar los conjuntos de entrenamiento de un algoritmo de análisis de componentes principales (PCA), y así obtener unos resultados más robustos y con un índice de fiabilidad elevado respecto a un conjunto de entrenamiento estándar.

Abstract

In today's society, almost globally connected, there is a growing need to determine or verify the identity of people. This need is evident in many aspects of daily life where authorization is required for access: from the presence control in a company, to access to boarding an aircraft.

Despite the recent technological advances, the identification and authentication of people is a challenge when it has to be done automatically, with accuracy and non-repudiation. This is because current methods based on passwords and identification cards with photograph, have proven to be ineffective when are subjected to the strong current demand of some specific environments.

The purpose of this paper is to explore the field of facial identification of people using multimodal analysis techniques; in other words, using RGB and depth images (3D), with the final aim of reviewing some of the techniques used in face recognition, as well as deepen in information systems that incorporate three-dimensional face detection and identification algorithms.

Finally, we have designed a prototype that uses the comparison of facial feature points between two subjects and homography techniques to improve the training sets for a principal component analysis algorithm (PCA), and thus obtaining more robust results with a high reliability index, against a standard training set.

Índex

1. Introducció

La disciplina de la Visió per Computador (o Visió Artificial), tracta del desenvolupament d'algorismes que reprodueixen una de les capacitats d'inferència més importants del cervell humà: la d'extreure propietats del món extern mitjançant la imatge dels objectes que capta l'ull.

Aquestes propietats permeten una comprensió de la imatge en termes d'extracció de la seva informació simbòlica utilitzant models construïts amb l'ajuda de la geometria, la física, l' estadística i la teoria de l'aprenentatge.

Com a disciplina científica, la Visió Artificial comprèn la teoria que hi ha darrere dels sistemes artificials que extreuen informació de les imatges. Les dades d'una imatge poden prendre moltes formes, com ara seqüències de vídeo, vistes de càmeres múltiples o multidimensionals, etc.

Com a disciplina tecnològica, la Visió per Artificial cerca l'aplicació de les teories i models a la construcció de sistemes de visió per computador. Alguns exemples d'aquestes aplicacions poden ser sistemes per:

- Identificació i autentificació.
- Detecció d'esdeveniments (p.e. vigilància)
- Modelat d'objectes o entorns.
- Interacció amb essers humans.
- Inspecció automàtica (p.e. en cadenes de producció)
- Etc.

Amb referència a la primera de les aplicacions esmentades, en la societat actual (que fa servir un model d'interconnexió gairebé global) existeix una necessitat creixent de determinar o verificar la identitat de les persones. Aquesta necessitat es fa patent en molts aspectes de la vida quotidiana on es requereix una autorització per l'accés: des del control de presència en una companyia, fins a l'accés a l'embarcament d'una aeronau.

Malgrat els darrers avenços tecnològics, la identificació i autentificació de persones esdevé un repte important quan aquesta s'ha de fer de manera automàtica amb precisió i no repudi, ja que els mètodes actuals basats en contrasenyes i targetes identificatives amb fotografia esdevenen ineficaços quan se'ls posa davant de la forta demanda actual en alguns entorns concrets.

Els mètodes tradicionals d'identificació, es poden agrupar en tres classes diferenciades: Possessions, on hom s'identifica mitjançant quelcom que té (per exemple una targeta xip); coneixement, on la identificació es fa per quelcom que es sap (per exemple una clau numèrica); i per últim biometria, on hom s'identifica per alguna característica pròpia com a individu (per exemple, la seva impremta dactilar)

En aquest treball incidirem en els aspectes de la Biometria aplicada a la identificació de persones, mitjançant el reconeixement dels seus principals *identificadors* facials

(comparant-los amb els recollits prèviament en una base de dades), utilitzant tècniques de visió per computador

2. Objectius

L'objectiu principal d'aquest treball és l'estudi dels diferents algorismes i tècniques emprades pel reconeixement facial dins de la disciplina de la Visió per Computador i la seva extensió a l'anàlisi multimodal de dades RGB-Depth.

Un altre objectiu, el secundari, és el dissenyar i desenvolupar diferents prototipus de programari que permetin aprofundir en els conceptes relacionats amb la identificació de persones. Per assolir aquest segon objectiu, es faran servir tècniques i recursos actuals utilitzats pel reconeixement facial, prioritzant l'ús de codi públic i llibreries de codi obert.

3. Estat de l'art

3.1. Antecedents

El focus d'atenció primari en les relacions socials és la cara, ja que és el principal element per transmetre la identitat d'una persona així com les seves emocions, el que permet el seu reconeixement. Els espectaculars avenços en la potència de càlcul dels ordinadors en les darreres dècades, han fet possible que es pugui fer un reconeixement similar, de manera automàtica.

Els primers algorismes que es van fer servir ja fa dècades, basats en models geomètrics simples, han evolucionat cap a una ciència de representacions matemàtiques complexes i processos de coincidència. Els darrers avenços i iniciatives han impulsat les tecnologies de reconeixement facial, de manera que avui en dia aquesta tecnologia està sent utilitzada per combatre el frau de passaports, suport a l'ordre públic, identificació de nens extraviats i minimitzar el frau en les identificacions

El primer sistema semiautomàtic va ser desenvolupat als anys 60. Per al reconeixement facial es requeria de l'administrador per localitzar trets (com a ulls, orelles, nas i boca) en les fotografies abans que aquest calculés distàncies a punts de referència en comú, els quals eren comparats després amb dades de referència.

Als anys 70 Goldstein, Harmon, & Lesk [8], van fer servir 21 marcadors subjectius específics tals com el color del cabell i grossor de llavis per automatitzar el reconeixement facial. El problema amb aquestes solucions prèvies era que es computaven manualment. El 1988 Kirby & Sirobich van aplicar anàlisi de components principals, una tècnica estàndard de l'àlgebra lineal, al problema del reconeixement facial. Això va ser considerat com una fita en mostrar que es requerien menys de 100 valors per xifrar encertadament la imatge d'una cara convenientment alineada i normalitzada [9].

El 1991 Turk & Pentland van fer servir la tècnica Eigenfaces mitjançant la qual l'error residual podia ser utilitzat per detectar cares en les imatges [5]. Això va suposar un descobriment que va permetre sistemes automatitzats de reconeixement facial en temps real fidedignes. Si bé l'aproximació era un tant forçada per factors ambientals, va crear un interès significatiu en posteriors desenvolupaments d'aquests sistemes.

La tecnologia inicialment va capturar l'atenció del públic a partir de la reacció dels mitjans a una prova d'implementació en el Super Bowl de la NFL al gener de 2001, la qual va capturar imatges de vigilància i les va comparar amb una base de dades de arxius fotogràfics digitals. Aquesta demostració va iniciar un anàlisi sobre com usar la tecnologia per satisfer necessitats nacionals, mentre es prenien en consideració les preocupacions socials i de privadesa del públic [1].

3.2. Biometria

El reconeixement biomètric respon a un sistema automàtic basat en la intel·ligència artificial i el reconeixement de patrons, que permet la identificació i/o verificació de la identitat de persones a partir de característiques morfològiques o de comportament, pròpies i úniques de l'individu, conegudes com *autentificadors*.

Com a principals *autentificadors* podem esmentar les empremtes dactilars, la geometria de la mà, la cara, el termograma facial, l'iris, la retina, la veu, l'estil d'escriptura, etc. Així mateix, la naturalesa del tipus de característica, morfològica o de comportament, es troba directament relacionada amb el grau de variació de les mateixes amb el pas del temps [4]

L'ús principal de la biometria és el reconeixement humà, el qual consisteix en la identificació i verificació de persones. Per identificació, el sistema biomètric respon a la pregunta "Qui és el subjecte X?". El sistema biomètric ha de llegir la mostra i la comparar-la amb les mostres emmagatzemades en una base de dades. Aquest tipus de comparació s'anomena cerca de "un-a-molts" (1:N). Així mateix, per a la verificació d'un usuari que diu posseir la identitat del subjecte X, el sistema ha de respondre a la pregunta de "És aquest el subjecte X?", fent un tipus de cerca de "un-a-un" (1:1). Prèvia a la cerca, l'usuari ha de proveir les dades biomètriques a verificar i el sistema identificarà o no al subjecte.

L'ús de les característiques biomètriques facials té molts avantatges, ja que el rostre és tret distintiu mes comú i més acceptat per totes les persones per reconèixer-se, a més de que el mètode d'adquisició de les imatges facials és no intrusiu. En aquest treball ens centrarem en l'ús de les característiques biomètriques facials per a la identificació de subjectes.

3.3. Conceptes bàsics

El problema del reconeixement facial pot ser formulat de la manera següent: donada una imatge d'entrada i una base de dades de persones conegudes, com podem verificar o determinar la identitat de la persona referenciada per la imatge d'entrada?

Per a resoldre aquest problema, es defineix un procés de reconeixement facial, que consta de quatre fases principals, que podem descriure com:

Detecció: es detecta que hi ha una cara a la imatge, sense identificar-la. Si es tracta d'un vídeo, també podem fer un seguiment de la cara. En aquesta fase es facilita la localització i l'escala a la qual trobem la cara.

Alineació: es localitzen els components de la cara i, mitjançant transformacions geomètriques, es normalitzen respecte a propietats geomètriques, com la grandària i la posició, i propietats fotomètriques, com ara la il·luminació.
Per normalitzar les imatges, es poden seguir diferents regles, com la distància entre les pupil·les, la posició del nas o la distància entre les comissures dels llavis. També s'ha de definir la grandària de les imatges i la gamma de colors emprada.

Extracció de característiques: s'extrau la informació necessària per tal de distingir entre les cares de diverses persones segons variacions geomètriques o fotomètriques. Es crea un vector que servirà per emmagatzemar les característiques a una base de dades, o bé per reconèixer una persona que hi figuri.

Reconeixement: el vector de característiques extret es compara amb els vectors de característiques emmagatzemats a la base de dades. Si es troba una coincidència amb un percentatge elevat de similitud ens retorna una identificació positiva. Altrament, el retornat és un negatiu.

3.4. Avantatges i desavantatges

L' avantatge principal de l'ús de la Biometria resideix en que la informació és única per a cadascun dels individus i que això permet la seva identificació individual malgrat els canvis produïts pel pas del temps.

Tenint en compte els pilars de la seguretat (autenticació, privacitat -confidencialitat de dades-, autorització -control d'accés-, la integritat i el no repudi), la Biometria, en general, és una tècnica que pot proporcionar el requeriments necessaris amb una fiabilitat molt alta. Tot i això, malgrat que les tècniques biomètriques estan considerades en general com les més efectives i segures per a la identificació (en alguns casos són molt difícils de falsificar), hem de considerar les avantatges i desavantatges de cadascun dels mètodes disponibles, de manera individual per a resoldre cada situació en concret. En el cas estudiat, una enumeració dels avantatges i els desavantatges que poden presentar les tècniques de reconeixement facial pot ser:

Avantatges

- No és invasiva
- Tecnologia relativament econòmica
- Es pot fer servir en escenaris on el contacte físic és difícil.
- Evita el contacte amb dispositius d'us públic, el que permet una asèpsia convenient en alguns entorns controlats (p.e. hospitals)

- Es pot fer servir en escenaris on la cooperació del subjecte no sigui obligatòria, però que aquest sigui conscient de que està sent registrat.

- Es pot usar en escenaris on el subjecte no és conscient que està sent reconegut (vigilància i sistemes de seguretat)

Desavantatges

Encara cal resoldre una sèrie d'inconvenients per tal de fer una correcta implantació de les tècniques de reconeixement facial. Aquests inconvenient poden ser, entre altres:

- Orientació del rostre. Els sistemes actuals són capaços de reconèixer el rostre d'una persona de front o de perfil. Petites variacions de l'angle del cap respecte a la càmera afecten al reconeixement, reduint la seva efectivitat.

- Il·luminació. Els canvis d'il·luminació poden afectar al rendiment del sistema.

- Oclusió a causa d'objectes o accessoris tals com a ulleres de sol, barrets que poden ocultar part de les característiques facials del subjecte, dificultant el seu reconeixement.

- Expressió facial. L'expressió de la cara pot canviar la percepció del sistema sobre la cara analitzada, per exemple, badallant la grandària de la boca és molt major que en condicions normals.

- Complexitat dels algorisme s: Malgrat que el reconeixement facial és la més natural de les tecnologies de reconeixement biològic, tenint en compte les regles cognitives dels éssers humans, el seu algorisme és deu vegades més complex que, per exemple, un algorisme d'identificació d'empremtes dactilars.

4. Tècniques de reconeixement Facial. Mètodes i algorismes

Actualment la investigació en reconeixement facial és madura i s'han desenvolupat diversos mètodes amb els que s'aconsegueix major o menor efectivitat, però gairebé tots es poden agrupar en quatre famílies principals:

- Mètodes de comparació holística, els quals usen la imatge completa com a entrada al sistema de reconeixement. Els mes utilitzats són el PCA (Principal Component Analysis, més conegut com a *eigenfaces*) descrit per M. TURK I A. PENTLAND [7], i el LDA (Linear Discriminant Analysis)[6].
- Mètodes de comparació basats en l'extracció de característiques, els quals necessiten separar les característiques facials locals com a ulls, boca, nas, etc. per alimentar un classificador. S'utilitzen tècniques estadístiques de reconeixement de patrons per a la identificació (R.BRUNELLI I T. POGGIO [12]).
- Mètodes basats en models, que construeixen un model descriptiu de la cara humana, amb el que es poden detectar amb precisió les diverses variacions facials.
- Altres mètodes (híbrids), els quals són una barreja dels anteriorment citats.

A la figura 1, es pot observar un arbre esquemàtic amb els diferents escenaris actuals i els mètodes associats.

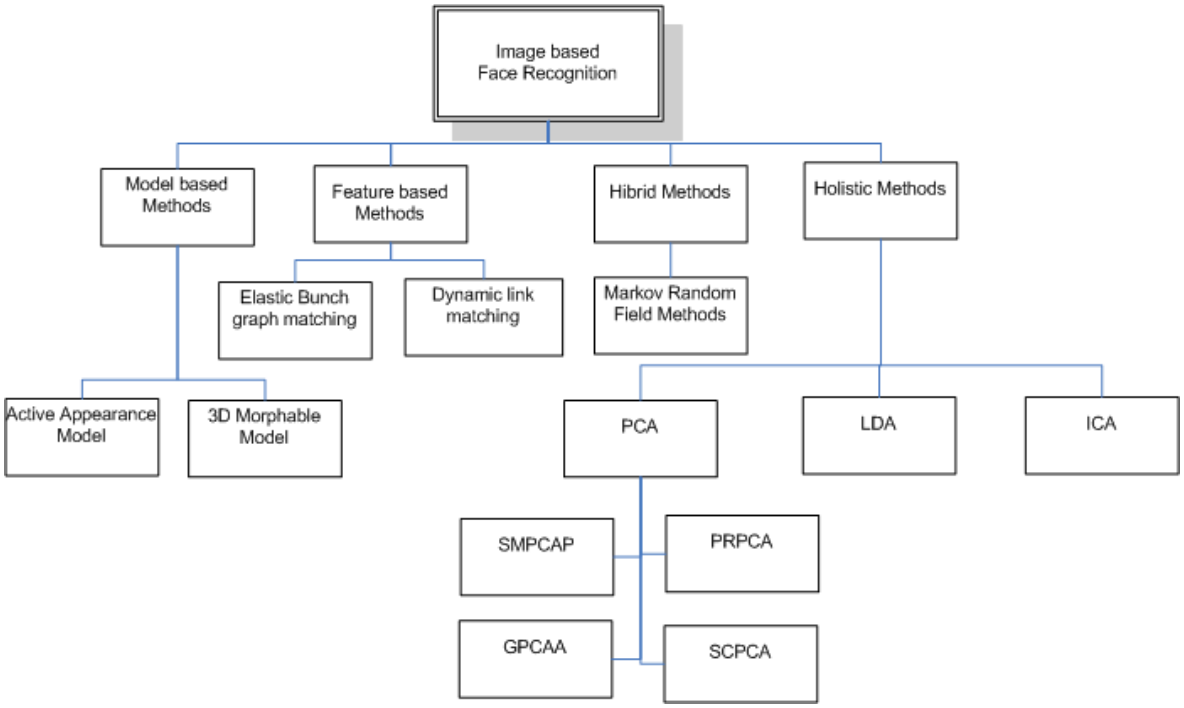


Figura 1: Mètodes utilitzats per a la classificació.

Malgrat que no és l'objectiu d'aquest estudi fer un anàlisi exhaustiu de tots els mètodes i algorismes utilitzats avui en dia per al reconeixement facial, és convenient, a mode d'introducció, descriure els algorismes de classificació més comuns en cadascun dels mètodes utilitzats.

4.1. ADABOOST

Adaboost és un algorisme d'aprenentatge automàtic desenvolupat per Y. Freund i R. Schapire [13], basat en el treball sobre detecció de cares proposat per Viola & Jones [14]. Adaboost és molt utilitzat a causa de la seva robustesa i alta velocitat. Fa servir detectors basats en una cascada de classificadors que és explorada per tota la imatge en múltiples escales i localitzacions.

Cada etapa de la cascada es basa en l'ús de característiques simples tipus Haar (eficientment computades utilitzant la imatge integral) seleccionades i combinades mitjançant Adaboost durant l'entrenament. L'eficiència d'aquest esquema resideix en el fet que els negatius (la immensa majoria de les finestres a explorar) van sent eliminats progressivament, de manera que les primeres etapes eliminen a un gran nombre d'ells (els més fàcils) amb molt poc cost de processament. Això permet que les etapes finals tinguin temps suficient per encarregar-se de classificar correctament els casos més difícils.

4.2. PCA (Principal Component Analysis)

Respecte als mètodes holístics, el algorisme més utilitzat es el conegut com a PCA o Eigenfaces, que consisteix en un sistema d'identificació en quasi temps real, que fa un seguiment del cap del subjecte i intenta el seu reconeixement mitjançant la extracció de diverses característiques de la cara. Després les compara amb altres de ja conegudes, emmagatzemades a una base de dades.

Aquest algorisme planteja la identificació facial com a un problema de reconeixement en dues dimensions, basant-se en que les fotografies de les cares són normalment frontals , i per això, les seves característiques poden ser descrites per un conjunt de vistes en 2D.

Amb Eigenfaces, la imatge a analitzar i la galeria d'imatges han de ser de la mateixa grandària i han d'haver estat normalitzades prèviament per tal d'alinear els ulls i boques dels subjectes a les imatges. L'aproximació de Eigenfaces és després utilitzada per reduir la dimensió de les dades per mitjà de fonaments de compressió, cosa que ens dona l'estructura de baixa dimensió mes efectiva dels patrons facials.

Aquesta reducció en les dimensions descarta informació que no és útil [10] i descompon de manera precisa l'estructura facial en components ortogonals (no correlatius) coneguts com a Eigenfaces. Cada imatge facial pot ser representada com una suma ponderada (vector identificatiu) dels Eigenfaces, els quals són emmagatzemats en un conjunt d'una dimensió.

Matemàticament parlant, el que es desitja amb aquesta técnica és trobar les components principals d'una distribució de rostres, o el que a partir d'ara es denominarà com eigenvectors de la matriu de covariança d'un conjunt d'imatges de cares, tractant a una imatge com a un vector en un espai multidimensional [B].

Els eigenvectors poden ser interpretats com a un conjunt de característiques, que totes juntes, caracteritzin la variació entre les imatges de les cares. Cadascuna de les localitzacions d'una imatge contribueix en major o menor mesura a cada eigenvector, pel que pot mostrar-se al eigenvector com a una representació de la cara anomenada eigenface.

Cadascuna de les eigenfaces, es desvia del gris uniforme en aquell punt on una característica inicial és diferent del conjunt de cares d'entrenament. En conseqüència, l eigenfaces són una mena de mapa de les variacions entre imatges.

L'avantatge principal d'aquesta tècnica és que pot reduir les dades necessàries per identificar l'individu a 1/1000 de les dades presentades. A la figura 2 podem veure una detall d'eigenfaces estàndard d'una cara. Els vectors identificatius són derivats utilitzant Eigenfaces [11].

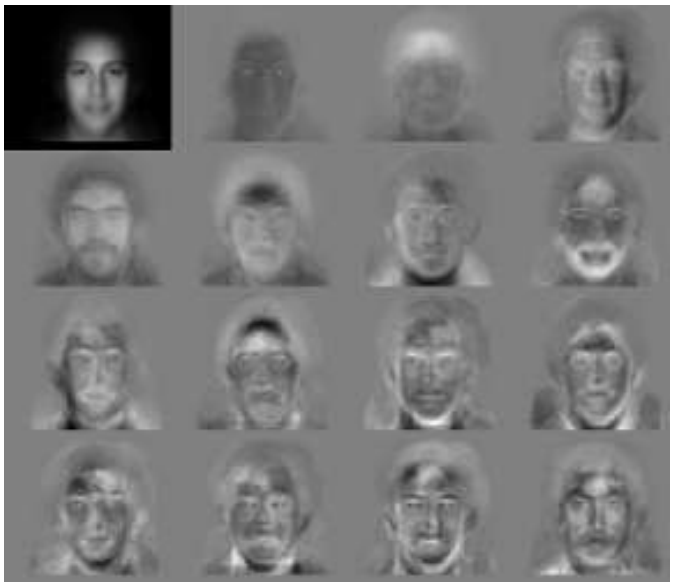


Figura 2: Eigenfaces Estàndard (MIT).

El procés de reconeixement facial utilitzant Eigenfaces i descomposició en valors principals (PCA) consta de les fases següents:

- Emmagatzemament d'un conjunt d'imatges (base de dades d'imatges) i entrenament. És convenient que aquesta base de dades consti de subconjunts d'imatges per a cada persona, i que contingui diferents posis i condicions d'il·luminació.

- Càlcul dels eigenfaces del conjunt d'entrenament, emmagatzemant únicament les M imatges que corresponguin amb els *eigenvalues* majors.

Aquestes M imatges defineixen l'espai de cares. Si s'afegeixen noves cares al sistema, les eigenfaces han de ser actualitzades o re calculades.

- Càlcul de la corresponent distribució en l'espai de pesos M-dimensional per a cada individu conegut mitjançant la projecció de la seva cara sobre l'espai de cares.

Una vegada completats els passos anteriors, pel reconeixement de noves cares s'haurà de:

- Calcular un conjunt de pesos basats en la imatge d'entrada i les *M-eigenfaces* mitjançant la projecció de la imatge d'entrada sobre cadascuna de les eigenfaces.
- Determinar si la imatge de la cara pertany o no al conjunt veient el proper component que està a l'espai de les cares.
- Opcionalment, les *eigenfaces* i/o els pesos patrons es podrien actualitzar, i si la mateixa cara desconeguda és vista diverses vegades, es podria afegir a la base de cares conegudes creada en el procés d'entrenament.

4.3. LDA (Linear Discriminant Analysis)

Originalment desenvolupat el 1936 per R. A. Fisher [15], l' anàlisi discriminant és un mètode clàssic de classificació que ha resistit la prova del temps. L'anàlisi discriminant sovint produeix aproximacions que s'aproximen en precisió (i de vegades la superen) a la dels mètodes moderns més complexos.

L'esquema LDA permet utilitzar la informació entre membres de la mateixa classe per desenvolupar un conjunt de vectors de característiques, on les variacions de les diferents cares s'emfatitzen, mentre que els canvis a causa de les condicions d'il·luminació, expressió facial i orientació de la cara no ho fan. A diferència de PCA, que és un mètode de reducció dimensional sense supervisió, LDA sí és supervisat i utilitza informació de les dades.

El sistema LDA [16] intenta maximitzar la variància de les mostres entre classes, i al mateix temps, minimitzar la variància entre mostres de la mateixa classe. Formalment això s'aconsegueix construint dues matrius de dispersió a partir de les mateixes imatges d'entrenament. En una de les quals es representa la dispersió entre les diferents classes (imatges de persones diferents) i en l'altra es representa la dispersió entre membres de la mateixa classe (imatges d'una mateixa persona).

A la figura 3 podem veure uns exemples de fotografies originals a l'esquerra; al centre la màscara estàndard aplicada, i a la dreta els exemples després del preprocessament.

Figura 3: Exemples de la base de dades FERET.

Matemàticament parlant, LDA té com a objectiu la conversió d'un problema d'alta dimensionalitat en un de baixa. Per això LDA projecta les dades en un espai vectorià de baixa dimensionalitat, de manera que la distancia entre classes i la distancia entre la mateixa classe es maximitza, garantint que la discriminació entre les classes és la màxima possible.

El mètode LDA obté millors resultats que PCA en cas que existeixin variacions en les condicions d'il·luminació dels subjectes i variacions en el gest i la posi.

4.4. EBGM (Elastic Bunch Graph Matching)

Respecte als mètodes basats en característiques, podem destacar el algorisme EGBM [18] o de correspondència entre agrupacions de grafs elàstics, que té en compte que les imatges facials reals tenen característiques no lineals que no són tractades en altres mètodes d'anàlisis lineals, com ara variacions de la il·luminació, posi i de l'expressió de la cara.

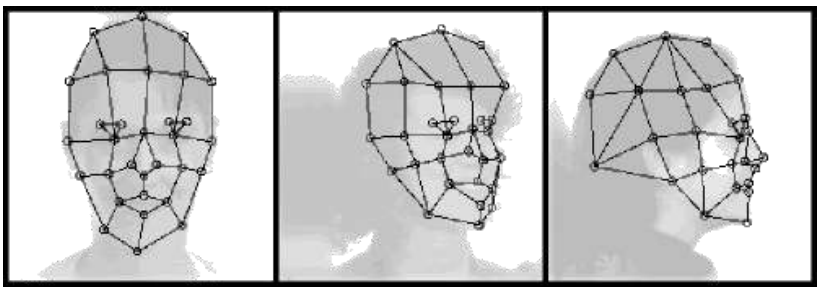


Figura 4: Correspondència entre agrupacions i grafs elàstics.

La tècnica EBGM té bàsicament dues etapes: la primera consisteix a ajustar un graf de punts principals a la cara de l'individu, utilitzant per a això un model estadístic d'aquest graf; la segona etapa extreu característiques locals en aquests punts i troba la distància del graf obtingut i els seus descriptors al graf emmagatzemat de la persona a identificar. Depenent de la distància calculada es ratifica o no la identitat de l'individu.

4.5. Modelat 3D

Els mètodes basats en models 3D [18] intenten obtenir característiques biomètriques de diverses imatges per tal d'obtenir el reconeixement (com ara distancia entre ulls, gruix del nas, etc.). Es tracta d'aconseguir algorismes de reconeixement facial robusts en front dels canvis de l'entorn (il·luminació, posi, etc.) amb l'objectiu final de reconstruir un model el més descriptiu possible de la cara humana, que representi amb precisió les variacions facials del subjecte analitzat.

El procés que segueixen aquests mètodes està format bàsicament per tres passes:

- Construcció del model.
- Ajust del model a la imatge de test.
- Utilització del model ajustat per calcular la similitud entre la imatge de test i les imatges de referència, el que donarà, si es cau, el reconeixement.

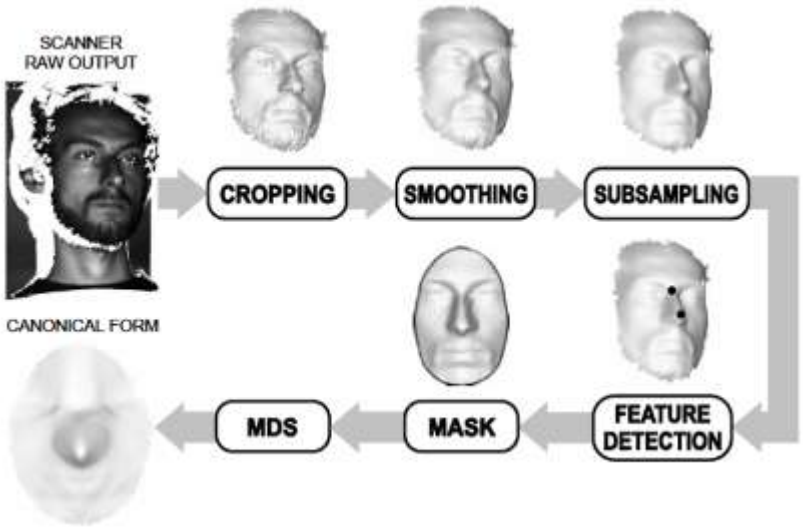


Figura 5: diagrama de funcionament d'un sistema de reconeixement basat en 3D.

5. Preliminars del projecte

L'algorisme proposat pel prototipus funcional d'aquest projecte identificarà un conjunt de subjectes de test, a partir de tècniques de processament PCA, combinades amb classificació basada en SVM (Support Vector Machines) i reforçades amb un tractament previ de les imatges de la base de dades d'entrenament: Escalat, equalització i Homografia (utilitzada per a la transformació de les diferents imatges d'entrenament respecte a la imatge frontal de cada subjecte individual).

5.1. Llibreries utilitzades

Per tal de desenvolupar aquest projecte, s'ha utilitzat el darrer SDK de Microsoft pel dispositiu Kinect, a més d'un conjunt de llibreries Open Source i de lliure distribució. En concret, OpenCV pel tractament d'imatges d'alt nivell, OpenNI per a interactuar amb els dispositius de captura d'imatge utilitzats i, per últim PCL, utilitzada per treballar amb imatges 2D i 3D, i pel processament de núvols de punts.

A continuació es detallen les característiques principals de cadascuna d'elles.

5.1.1. OpenCV

OpenCV (Open Source Computer Vision Library) és una llibreria oberta desenvolupada per Intel que proporciona un alt nivell funcions per al processament d'imatges. Aquestes funcions permeten als programadors crear aplicacions molt potents en el domini de Visió Artificial. OpenCV ofereix molts tipus de dades d'alt nivell com jocs, arbres, gràfics, matrius, etc.

OpenCV implementa una gran varietat d'eines per a la interpretació de la imatge. És compatible amb Intel Image Processing Library (IPL) que implementa algunes operacions primitives en imatges digitals com ara binarització, filtrat, estadístiques de la imatge, piràmides, etc.

Open CV és multi plataforma. Existeixen versions per a GNU / Linux, Mac OS X i Windows. Conté més de 500 funcions que abasten una gran gamma d'àrees com reconeixement d'objectes (inclòs el reconeixement facial), calibratge de càmeres, visió estèreo i visió robòtica.

OpenCV es fa servir per desenvolupar:

- Sistemes de seguretat
- Sistemes d'inspecció per a la cadena de producció
- Anàlisis mèdic d'imatges
- Detecció de cares i identificació
- Etc.

Malgrat que OpenCV és la llibreria més utilitzada degut al seu caràcter obert i a la seva portabilitat, existeixen altres bones llibreries també disponibles com:

1. AForge.NET – Llibreria de visió per computador per .NET framework.
2. VXL - The Vision-*something*-Libraries.
3. IVT - Integrating Vision Toolkit.
4. VideoLab – Conjunt de components per processament de vídeo.

5.1.2. OpenNI

El terme d'interacció Natural (en anglès, NI) es refereix a un concepte on la iteració home-dispositiu es basa en els sentits humans, centrant-se principalment en l'audició i visió.

OpenNI (Open Natural Interaction) és un entorn de treball multi idioma i multi plataforma que defineix interfícies per escriure aplicacions que utilitzen Interacció Natural. L'objectiu principal de OpenNI és formar una API estàndard que permet la comunicació amb :

- Visió i sensors d'àudio (dispositius que "veuen" i "escolten" les figures i els seus voltants.)
- Visió i la percepció d'una capa intermèdia (middleware) d'àudio (els components de programari que analitzen l'àudio i les dades visuals que es graven des de l'escenari, i comprendre'ls). Per exemple, el programari que rep les dades visuals, com una imatge, retorna la ubicació del palmell d'una mà detectant-la a la imatge.

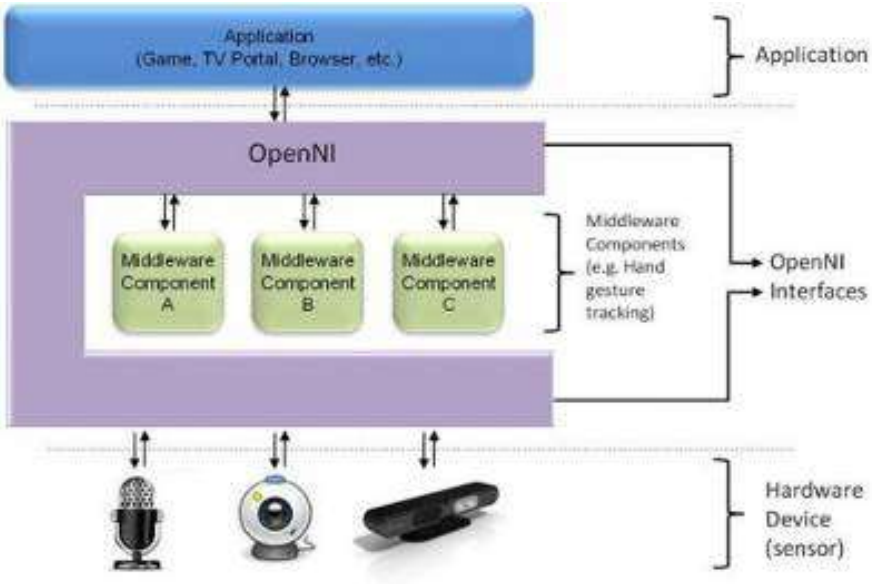


Figura 6: Capa intermèdia (Middleware) OpenNI.

OpenNI proporciona un conjunt de APIs per ser implementades pels dispositius amb sensor, i un conjunt de APIs que són implementats pels components de middleware. En trencar la dependència entre el sensor i la capa intermèdia de programari , OpenNI API permet a les aplicacions ser escrites i portades sense esforç addicional per operar en la part superior de diferents mòduls middleware ("escriure una vegada, desplegar a tot arreu ").

5.1.3. PCL (Point Cloud Library)

PCL és un entorn Open Source per treballar amb imatges 2D i 3D i pel processament de núvols de punts. PCL ha estat desenvolupat per un consorci d'investigadors i enginyers de tot el món. Està escrit en C++ i cedit sota la llicència BSD. El conjunt de biblioteques que inclou, conté un gran nombre d'algorismes per filtrat, extracció de característiques, reconstrucció de superfícies, registre i segmentació.

Un núvol de punts és una sèrie de dades en un espai de qualsevol nombre de dimensions que es troben dispersos. Típicament parlem de núvol de punts en espais 2D/3D quan ens referim a les dades (profunditat, color, textura, forma) que obtenim mitjançant sensors (càmeres, làsers, sonars, etc.).

Habitualment les dades capturades en l'espai 3D són representades en un plànol 2D, d'aquesta manera si volem passar del plànol 2D a l'espai 3D, ens trobem amb què hi ha molts punts de l'espai 3D pels quals no disposem d'informació, de manera que a l'espai 3D només tenim informació d'alguns punts concrets. En aquest cas, aquests punts formen un núvol de punts.

Els algorismes de PCL es poden utilitzar, per exemple per a la percepció i filtrat dels valors extrems de dades amb molt soroll, pel tractament de núvols de punts 3D en conjunt, per segmentar les parts rellevants d'una escena, per extreure punts clau i calcular descriptors, per reconèixer objectes en el món sobre la base de la seva aparença geomètrica i per crear superfícies a partir de núvols de punts i visualitzar-les.

Figura 7: Exemple de captura de núvol de punts PCL.

5.1.4. Microsoft Kinect SDK

La darrera revisió del Kinect Sowtware Development kit per a Windows, permet als desenvolupadors utilitzar llenguatges d'alt nivell com c++, c# o VB.net, per crear aplicacions que suporten reconeixement de gestos i veu, mitjançant la utilització del sensor Kinect en un ordinador amb sistema operatiu Windows 7, Windows Embedded Standard 7 o superior. Addicionalment, el SDK inclou APIs i interfícies per a dispositius varis.

El SDK està orientat a la investigació acadèmica principalment encara que també a programadores particulars amb l'objectiu que experimentin amb la creació de interfícies naturals d'usuari.

Com a funcions principals, el SDK permetrà:

- Skeletal Tracking d'una o dues persones que estiguin en l'angle de visió de Kinect.
- Calcular la distància dels objectes al sensor Kinect mitjançant la seva càmera infraroja que proporciona informació de profunditat.
- El processament avançat d'àudio mitjançant la seva matriu de quatre micròfons.

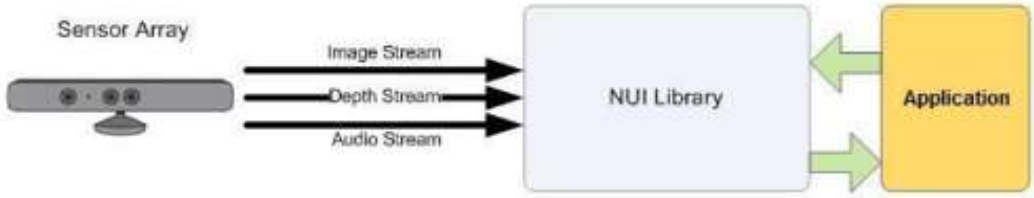


Figura 8: Interacció de la capa de hardware i software amb les aplicacions.

Respecte a la revisió anterior (anomenada Beta 2) i les precedents, el SDK ha estat actualitzat per suportar més prestacions, proporcionant als desenvolupadors més flexibilitat i un control més gran sobre el desenvolupament de les seves aplicacions. Am la nova versió, per exemple, els desenvolupadors poder emmagatzemar sessions d'usuari i fer-les servir més tard en automatització de tests, anàlisi de rendiment i escenaris basats en casos d'ús, així com crear estàndards de medició (benchmark) per a les seves aplicacions, amb la consegüent optimització de temps i recursos.

Addicionalment, la versió disponible del SDK, es complementa amb un paquet anomenat "Kinect for Windows developer toolkit" que inclou components reutilitzables, llibreries, eines addicionals i nous exemples.

NUI API

Aquesta és la principal API de Kinect per Windows. Dóna suport a la gestió de dades així com al control d'algunes de les propietats del dispositiu:

- Accés als diferents Kinect que estan connectats a un dispositiu.
- Accés als fluxos d'imatges i de dades de profunditat captats pels sensors de Kinect.
- Gestió de les imatges processades i dades de profunditat per donar suport al Skeletal tracking.

Aquesta API està dividida en quatre subsistemes. En inicialitzar la API haurem de seleccionar quins dels subsistemes utilitzarem i com ho farem:

- Color, l'aplicació envia les dades de la imatge captada pel sensor.
- Depth, envia les dades de profunditat captats pel sensor.
- DepthandPlayerIndex, retorna les dades de profunditat i si el píxel es correspon amb un jugador (subjecte) retorna l'índex del jugador.
- Skeleton, retorna les dades esquemàtiques dels punts d'unió d'articulacions del subjecte.

5.2. Entorn de treball

5.2.1. Maquinari

Maquinari de desenvolupament

- Sistema operatiu: Windows 7 (32 o 64 bits).
- Ordinador amb processador dual-core 2.66 GHz (mínim).
- Targeta gràfica compatible Windows 7
- Microsoft DirectX 9.0c.
- 2Gb de memòria RAM mínim.
- Dispositiu Kinect.

Microsoft Kinect

El sensor de Kinect és un dispositiu d'i molt complex, dissenyat permetre una interacció home-màquina d'alt nivell. Consta de:

- una càmera RGB,
- un sensor de profunditat i
- un micròfon multi-matriu bidireccional

El sensor de Kinect adquireix imatges de vídeo amb un sensor CMOS de colors a una freqüència de 30 Hz, en colors RGB de 32-bits i resolució VGA de 640×480 píxels. El canal de vídeo monocrom CMOS és de 16-bit, resolució QVGA de 320×240 píxels amb fins a 65,536 nivells de sensibilitat.

Figura 9: Dispositiu Kinect.

Per calcular distàncies entre un cos i el sensor, el sensor emet un feix làser infraroig que projecta un patró de punts sobre els cossos, amb el que la distància dels quals es pot calcular. Una càmera infraroja capta aquest patró i per maquinari calcula la profunditat de cada punt. El rang de profunditat del sensor de Kinect està entre 0.4 i 4 mts.

La matriu del micròfon té quatre càpsules, i opera amb cada canal processant en 16-bit d'àudio amb un ràtio de freqüència de mostra de 16 KHz. La càmera de Kinect funciona amb maquinari i programari propis per al reconeixement d'imatge.

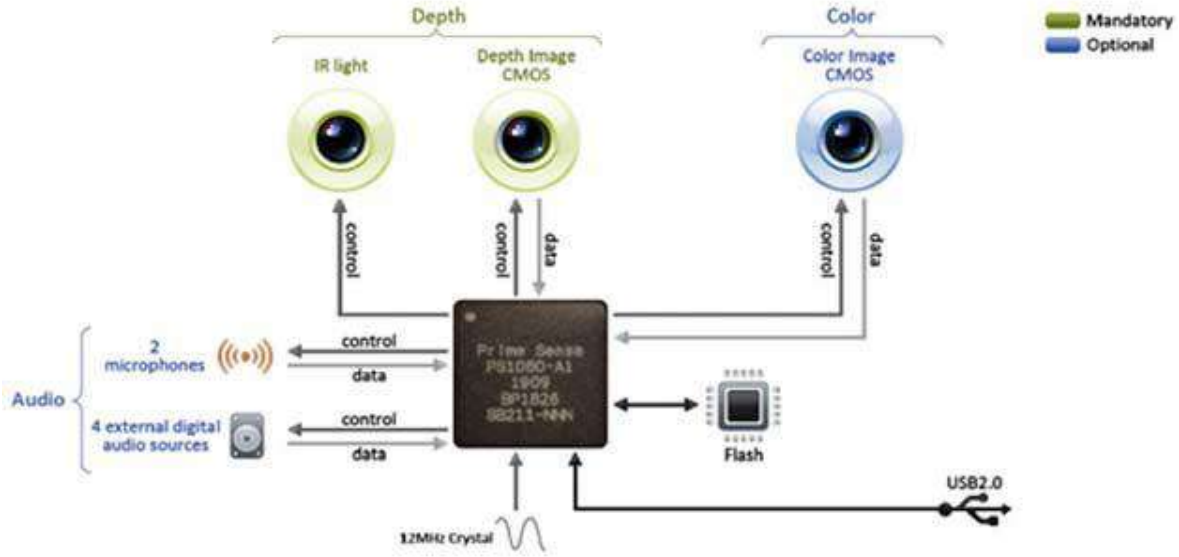


Figura 10: Composició interna del dispositiu Kinect.

La càmera i el projector de llum infraroja de Kinect, compten a més amb un firmware i un processador que utilitza algorismes per processar les imatges tridimensionals

La càmera té dues funcions principals:

- Generar un mapa en 3D de la imatge que té en el seu camp visual i
- Reconèixer humans en moviment entre els objectes de la imatge a partir de diferents segments de les articulacions del cos i un esquema en escala de grisos.

En un ampli camp visual amb objectes, la càmera Kinect tracta de reconèixer a quina distància estan els diferents objectes, distingint moviments en temps real. El sensor Kinect pot arribar a distingir la profunditat de cada objecte amb una resolució d'1 mil·límetre i les estimacions de l'altura i amplària amb una exactitud d'aproximadament 3 mil·límetres.

El processador és capaç d'interpretar els moviments que es registren dels objectes capturats per la càmera de Kinect en "esdeveniments amb significat" que apareixen en pantalla. Els moviments buscats per l'algorisme són contextualitzats. Per exemple, si s'està aplicant el sensor Kinect a un joc com Kinect Adventures, on un rai descendeix pel corrent del riu, atès que aquest joc requereix moviments tals com ajupir-se o tombar-se, l'algorisme buscarà la identificació d'aquests moviments en temps real per produir esdeveniments en pantalla. Si l'usuari navega pel menú amb interfase gràfica de Netflix llavors es buscaran moviments amb les mans horitzontals i verticals que seran registrats en els fenòmens de pantalla.

5.2.2. Programari

Per tal d'executar les aplicacions desenvolupades amb el SDK de Kinect serà necessari utilitzar un entorn natiu de Windows, això vol dir que no és possible executar les aplicacions en una màquina virtual ja que tant les llibreries compilades de Kinect com el SDK han d'estar instal·lats en la computadora on està corrent l'aplicació.

Els llenguatges utilitzats pel SDK son C# o C++.

Requeriments programari:

- Microsoft Visual Studio 2010 (MS VS Express o qualsevol edició).
- Microsoft .NET Framework 4 (instal·lat amb Visual Studio).
- Pels exemples de Skeletal amb C++:
- DirectX Software Development Kit (versión juny 2010 o posterior).
- DirectX End-User Web installer.

Pels exemples de veu:

- Microsoft Speech Platform – Server Runtime, versió 10.2 (per a x86).
- Microsoft Speech Platform- Software Development Kit, versió 10.2 (per a x86).
- Kinect for Windows Runtime Language Pack, versió 0.9.

6. Projecte de reconeixement facial

6.1. Diagrama de blocs

Els blocs principals que composaran el projecte es detallen al diagrama següent:

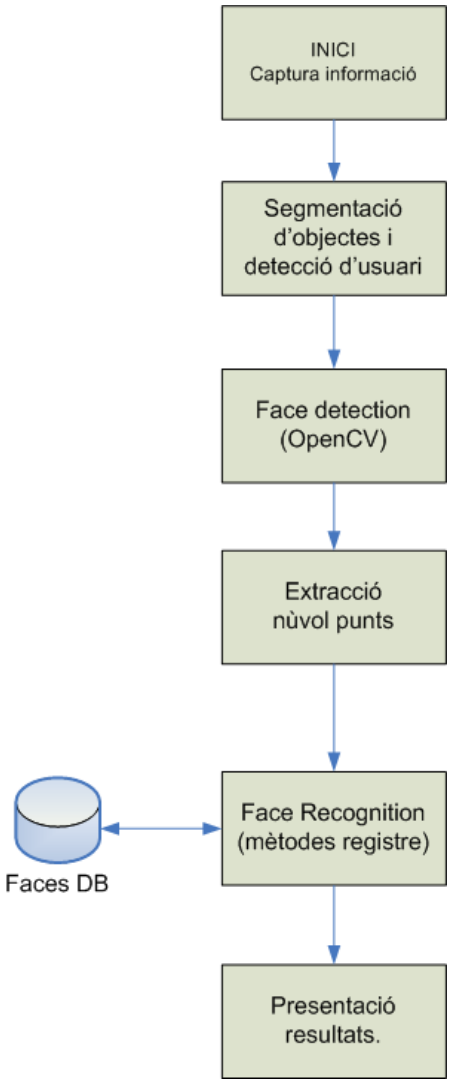


Figura 11: Diagrama de blocs del projecte.

La descripció detallada de cadascun d'aquests blocs es pot trobar en l'apartat següent.

6.2. Detall dels blocs principals

6.2.1. Captura d'informació

La captura de la informació, en aquest cas imatges estàtiques, es farà a partir del sistema de gravació en vídeo del dispositiu Kinect. Es gravaran imatges específiques del flux de vídeo (frames), en els intervals desitjats per tal d'aconseguir les imatges que mes tard formaran part de la base de dades així com les imatges de test per fer el reconeixement.

6.2.2. Segmentació d'objectes i detecció d'usuari

En aquesta fase, es segmenta el fons dels de les imatges utilitzades per l'algoritme, com a fase prèvia a la detecció de la cara dels diferents subjectes de la imatge subministrada. En el cas concret que pertoca al projecte d'identificació, es treballarà ama una imatge de subjecte únic i es realitzarà un tractament previ que consistirà en la separació del fons de la cara del subjecte i el retallat de la cara a l'àrea delimitada per la malla generada a partir de la informació 3D obtinguda del subjecte.

6.2.3. Detecció de la cara

En aquesta fase, el sistema detectarà si hi ha una cara a la imatge, sense identificar-la. En aquest bloc es facilita la localització i l'escala a la qual trobem la cara. La mecànica de reconeixement està basada en les característiques tipus Haar d'un objecte. Aquestes característiques ressenyen l'existència d'orientació del contrast entre regions dins d'una imatge. Un conjunt d'aquestes característiques pot ser utilitzada per determinar el contrast mostrat pel rostre humà i la seva interrelació espacial.

6.2.4. Extracció del núvol de punts i característiques

Mitjançant la utilització del entorn de desenvolupament de Microsoft SDK per a Kinect, aconseguim els dos vectors que representen els núvols de punts característics de la cara del subjecte. El primer vector obtingut, vector 2D, ens dona les coordenades X,Y dins de la imatge on es troben els 100 punts característics de la cara del subjecte, mentre que el segon vector obtingut, el 3D, ens dona informació sobre la rotació i translació de la cara. Aquest vectors de punts, es faran servir dins del projecte per tractar les imatges i per trobar quina es la vista de la cara amb més punts característics coincidents, abans de passar la imatge al sistema d'identificació.

6.2.5. Reconeixement facial

Utilitzant la imatge seleccionada a la fase anterior i la imatge de test del subjecte, es fa la crida l procés de reconeixement facial. Aquest procés cerca similitud entre les imatges passades mitjançant l'anàlisi de correlació de punts basat en un model PCA més SVM. El procés dona un índex de fiabilitat per a cada imatge de test i una predicció del subjecte, (candidat) corresponent a l'índex d'imatges de la base de

dades d'entrenament. El sistema presentarà la imatge seleccionada, juntament amb l'índex de probabilitat obtingut.

6.3. Prototipus preliminars

6.3.1. Captura d'informació amb Kinect

Per a la captura de la informació mitjançant kinect, és a dir imatges RGB i imatges depth amb informació de profunditat (3D) (necessàries per a la confecció de la base de dades d'imatges de treball), s'ha fet servir la implementació adaptada de la solució *NISimpleViewer* subministrada dins de la col·lecció d'exemples que incorpora OpenNI en el procés d'instal·lació.

Com a punt de partida, es crearà un model d'imatge tridimensional a partir d'imatges RGB i de profunditat obtingudes amb un dispositiu KINECT. Utilitzant tècniques de face tracking, deduirem la posició del cap del subjecte, els trets característics de les expressions facials i renderitzarem la cara del subjecte a l'hora que es crea una malla (grid tipus EGBM). D'aquesta malla podrem extreure les coordenades d'un conjunt de característiques facials del subjecte, que seran útils a l'hora de crear la base de dades d'imatges, prèvia a la detecció.

La funció utilitzada per a la gravació de les imatges depth obtingudes amb Kinect, en format OpenCV::Mat a un fitxer binari reutilitzable és la següent:

```
Mat depthImg(480,640,CV_16SC(1),(unsigned short*) pDepth);

// Grava matriu depth a binary file
int saveMat( const string& filename, const Mat& M){
    if (M.empty()){
       return 0;
    }
    ofstream out (filename.c_str(), ios::out|ios::binary);
    if (!out)
       return 0;
    int cols = M.cols;
    int rows = M.rows;
    int chan = M.channels();
    int eSiz = (M.dataend-M.datastart)/(cols*rows*chan);
    // Capcelera del fitxer
    out.write((char*)&cols,sizeof(cols));
    out.write((char*)&rows,sizeof(rows));
    out.write((char*)&chan,sizeof(chan));
    out.write((char*)&eSiz,sizeof(eSiz));
    // Dades de profunditat
    if (M.isContinuous()){
       out.write((char *)M.data,cols*rows*chan*eSiz);
    }
    else{
       return 0;
    }
    out.close();
    return 1;
}
```

Gravació d'imatges depth.

6.3.2. Detecció facial

Les llibreries OpenCV tenen tot un seguit de funcions orientades al reconeixement de patrons, sent el més característic el relacionat amb el reconeixement facial. Aquesta mecànica de reconeixement està basada en les característiques tipus Haar [5] d'un objecte.

Un procés de reconeixement pot ser molt més eficient si està basat en la detecció de característiques més significatives del tipus d'objecte que ha de ser detectat. Aquest és el cas de les característiques tipus Haar, que ressenyen l'existència d'orientació del contrast entre regions dins d'una imatge. Un conjunt d'aquestes característiques pot ser utilitzada per determinar el contrast mostrat pel rostre humà i la seva interrelació espacial.

La detecció d'objectes de OpenCV va ser inicialment proposada per Paul Viola i millorada per Rainer Lienhart [20]. Primer un classificador (en realitat una cascada de classificadors ampliats treballant amb característiques tipus Haar) és entrenat amb uns pocs centenars d'imatges d'exemple d'un objecte en particular (pot ser el mateix una col·lecció de cares com una col·lecció de cotxes), generant el que es coneix com a exemples positius, tots ells escalats a la mateixa grandària, així com exemples negatius (imatges arbitràries de la mateixa grandària).

La paraula "cascada" referida al classificador, ve donada perquè el resultat del classificador és el fruit de diversos classificadors molt més simples (etapes) que són aplicats subsegüentment a una regió d'interès fins que, en alguna etapa, el candidat és rebutjat o fins que es superen totes les etapes. La paraula "ampliats" es fa server degut a que els classificadors de cada etapa de la cascada són complexos en si mateixos i són construïts a part dels classificadors basics, usant una de les quatre diferents tècniques d'ampliació (votació per rellevància).

Després que el classificador hagi estat entrenat, pot ser aplicat a posteriori a una regió d'interès (d'igual grandària) sobre una imatge d'entrada. El classificador retorna un "1" si considera que hi ha alguna semblança entre l'hipotètic objecte de l'àrea d'interès i un "0" en cas contrari. Per tal de cercar l'objecte per tota la imatge, es pot moure la finestra de cerca al llarg de la imatge i comprovar cada àrea usant el classificador. Aquesta tècnica és coneguda com a *Windowing*.

El classificador ha estat dissenyat de manera que és fàcilment escalable; això es fa amb el propòsit que pugui trobar objectes d'interès a diferents grandàries, la qual cosa resulta molt més eficient que escalar la imatge d'entrada. D'aquesta manera, per trobar un objecte d'una grandària i una localització indeterminats en la imatge, el procés d' escanejat ha de ser executat diverses vegades a diferents escales.

Funcions CvHaarClassifierCascade I cvHaarDetectObjects

Per tal de dur a terme la detecció de cares s'ha utilitzat l'algorisme de Viola & Jones [14], que incorpora OpenCV, i que ens permet detectar diferents tipus d'objectes en imatges, utilitzant també els diferents tipus de classificadors que pot fer servir la llibreria.

En aquest cas utilitzem el classificador "haarcascade_frontalface_alt2.xml" que ens permet trobar cares que estiguin a la imatge i amb el qual podem obtenir un bon rendiment.

La implementació de les funcions classificador i de detecció d'objectes es pot apreciar en el fragment de codi següent:

```
//*************************************
// Classificador Haar
//*************************************
        CvHaarClassifierCascade*           cascade=           (CvHaarClassifierCascade*)cvLoad(
"haarcascade_frontalface_alt2.xml" );
    double scale = 1.3;

    static CvScalar colors[] = { {{0,0,255}}, {{0,128,255}},{{0,255,255}},   {{0,255,0}},
{{255,128,0}}, {{255,255,0}}, {{255,0,0}}, {{255,0,255}} };

//*************************************
// Detecció cares dins la imatge
//*************************************
    cvClearMemStorage( storage );
    CvSeq* objects = cvHaarDetectObjects( img, cascade, storage, 1.1, 4, 0, cvSize( 20, 20 ));

    CvRect* r;
    // Dibuixa rectangle en cares localitzades
    for( int i = 0; i < (objects ? objects->total : 0 ); i++ ){
        r = ( CvRect* )cvGetSeqElem( objects, i );
        cvRectangle( img, cvPoint( r->x, r->y ), cvPoint( r->x + r->width, r->y + r->height ),
            colors[i%8]);
    }
```

Funcions classificador i detecció d'objectes.

Resultats

El resultat obtingut amb la utilització de l'algorisme Viola & Jones[14] i el classificador *Haarcascade* es pot veure a la figura 12, on es pot veure que les cares de tots els subjectes de la imatge apareixen emmarcades per rectangles de color.

Figura 12: Resultats de l'aplicació de l'algorisme Viola & Jones.

6.3.3. Seguiment facial (face tracking)

Amb la utilització del Microsoft Face Tracking SDK de Kinect per Windows, juntament amb Kinect SDK es poden crear aplicacions que fan seguiment del rostre humà en temps real, amb relativa facilitat.

El component principal (core) del MS Tracking SKD, analitza la entrada d'informació rebuda directament de la càmera RGB del dispositiu Kinect, dedueix la posició del cap del subjecte i trets característics de les expressions facials, i posa aquesta informació a disposició de les aplicacions en temps real. Amb aquesta informació disponible, per exemple, podrem renderitzar la cara del subjecte i crear una malla (grid) que ens permetrà fer un seguiment de la cara mentre aquesta es desplaça per la imatge.

Sistemes de coordenades

El Face Tracking SDK utilitza el sistema de coordenades Kinect per a donar els seus resultats de seguiment en 3D seguint resultats. L'origen del sistema de coordenades es localitza al centre òptic de la càmera (sensor), l'eix Z està assenyalant a l'usuari, mentre que l'eix Y està assenyalant amunt (figura 13). Les unitats de mida són metres per translació i graus pels angles de rotació.

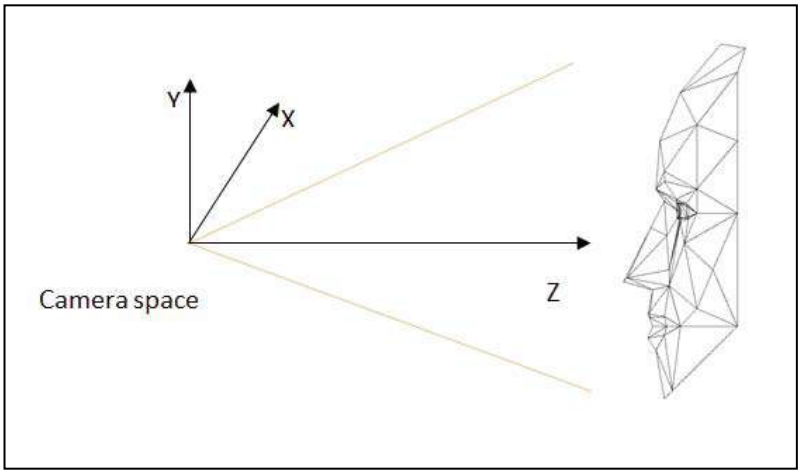


Figura 13: Sistema de coordenades de Kinect.

Les imatges d'entrada que utilitza el sistema de seguiment de Kinect utilitzen el sistema de coordenades 2D especificat a la figura següent. La qualitat del seguiment (tracking quality) es pot veure afectada per la qualitat de les imatges entrants (frames). Imatges poc il·luminades o difuses donaran resultats més pobres que les ben il·luminades i/o ben definides. Així mateix, les imatges en les quals les cares apareguin en una major grandària (més a prop) obtindran uns resultats millors en el seguiment que les que presentin cares més petites o llunyanes.

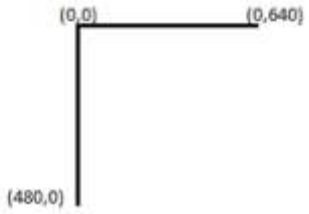


Figura 14: Eix de coordenades d'una imatge RGB.

Resultats del Face Tracking

Quan executem una funció de seguiment (StartTracking o ContinueTracking de l'SDK de Kinect), el sistema en retorna un objecte resultat (FTResult) que conté informació en temps real sobre la cara en seguiment. En concret, aquest objecte conté la següent informació sobre l'usuari:

- Tracking status (si el rostre del subjecte ha estat localitzat o no).
- Punts 2D de característiques corresponents a la cara del subjecte.
- 3D head pose (informació sobre la posició 3d del cap del subjecte).
- Aus (Animation Units).

Punts 2D

Un dels resultats que ens ofereix el sistema de seguiment, és un vector punts 2D (x,y) amb referència a la imatge RGB de la cara del subjecte. El nombre del punt, especifica la posició al vector 2D. A la figura següent, es poden apreciar 87 dels 100 punts del vector retornat.

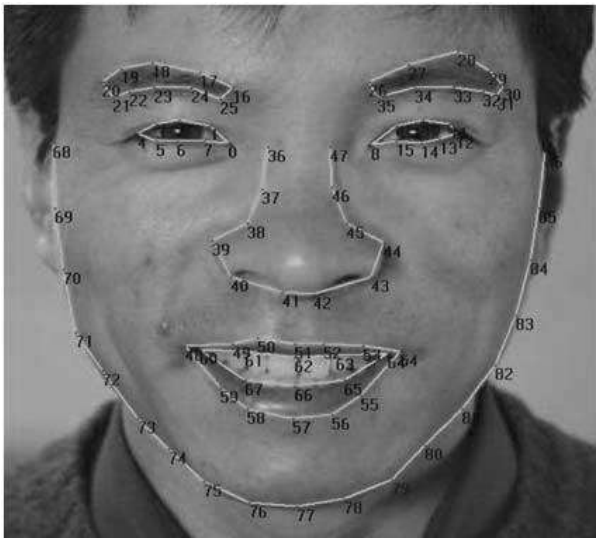


Figura 15: Punts de seguiment del sistema de tracking de Kinect.

A banda d'aquest punts mostrats a la imatge, existeixen 13 punts més que no es poden apreciar i que fan referència al centre dels ulls, els cantons de la boca i el centre del nas. El detall dels punts és es següent:

- 87: centre del ull esquerre.
- 88: centre del ull dret.
- 89: centre del nas.
- 90-94: cella esquerra des de l'interior.
- 95-99: cella dreta des de l'interior.

3D HEAD POSE

El sistema dona també informació de la posició espaial del cap del subjecte amb referència al sistema de coordenades 3D de Kinect especificat amb anterioritat.
En concret el sistema captura tres angles, anomenats en anglès *pitch, roll* i *yaw*, amb valors que van de -90º a +90º, sent 0º la posició neutral. El detall d'aquests angles es pot veure a la imatge següent:

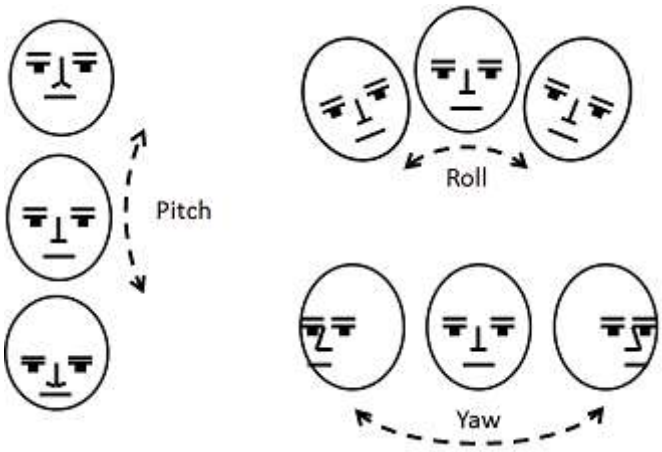


Figura 16: Posició 3D del cap del subjecte.

Animation Units (AU)

Per últim, els resultats d'una acció Face Tracking retonen una sèrie de valors expressats en termes de les sis unitats d'animació que són un subconjunt dels especificats als models Candide3 [1].

Implementació del Facial Tracking

La solució *FaceTrackingVisualization* pot ser utilitzada pel seguiment (tracking) de cares humanes amb una càmera Kinect connectada a un PC. El core de Face Tracking calcula les posicions 3D del punts de les diferents característiques facials del subjecte, així com la posició dins l'espai de la càmera del seu cap.

En aquest cas, la informació obtinguda es fa servir pel moure un avatar (que apareix a la part esquerra de la pantalla) i que segueix fidelment els moviments del subjecte una vegada s'estableix el seguiment.

Un exemple dels resultats obtinguts mitjançant la implementació de la solució *FaceTrackingVisiualization* es la imatge inferior, on es pot apreciar a la dreta de la imatge, en color Cian, el requadre de demarcació de la cara del subjecte, i en color grog la malla formada per triangles que uneixen els punts especificats en el vector 2D retornat pel sistema de seguiment de rostres de Kinect.

[1] CANDIDE és una parametrització de màscara facial concretament desenvolupada com a model de codificació de cares humanes. El seu baix nombre de polígons (aproximadament 100) permet una reconstrucció ràpida amb uns requeriments computacionals reduïts (http://www.icg.isy.liu.se/candide/).

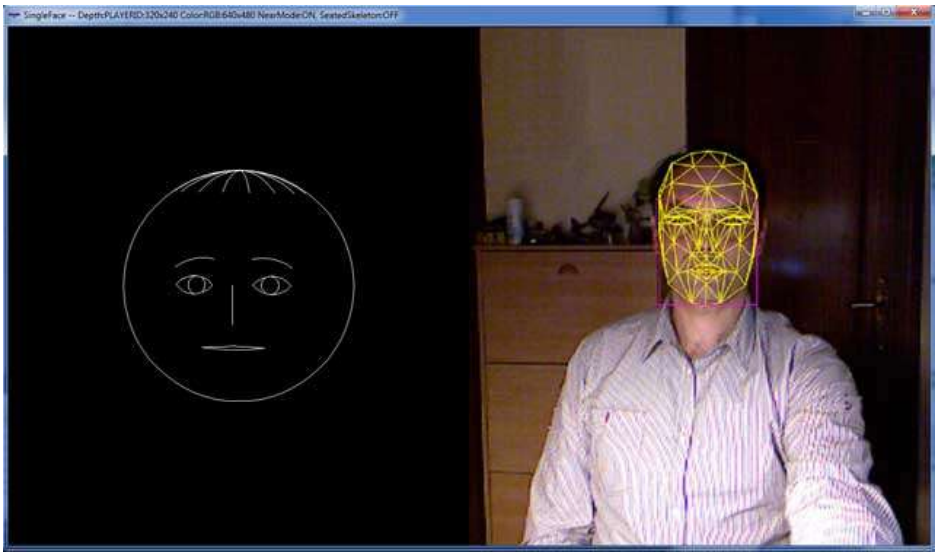


Figura 17: Resultats de la implementació de FaceTrackingVisualization.

En el projecte, la solució *FaceTrackingVisualization* ha servit de base per tal d'implementar la solució de captura d'imatges RGB segmentades i la recollida de les dades 2D i 3D per tal de crear la base de dades d'imatges.

Partint d'aquesta solució, s'han implementat les funcionalitats següents:

- Gravació en disc d'imatges rgb (bmp).
- Gravació en disc d'imatges rgb (compressió JPG).
- Gravació en disc de la imatge capturada, amb segmentació del fons, només amb la cara del subjecte visible.
- Gravació fitxer bin amb núvol de punts de Face 2D i rotació/translació 3D.
- Gestió de fitxers depth.

Extracció del núvol de punts de característiques facials

En aquest cas, no s'han fet servir les llibreries PCL per a la extracció del núvol de punts de característiques facials. El sistema de seguiment de Kinect, proporciona tota la informació de la malla facial necessària mitjançant els vectors 2D y la informació 3D continguda a l'objecte FTResult retornat. Per tal de passar aquesta informació a un fitxer binari que pugui esser gravat a disc, s'ha definit una estructura a partir de l'objecte obtingut del sistema de seguiment (amb les característiques facials del subjecte).

L'estructura dels paràmetres gravats al disc en format binari és la següent:

```
UINT numAU;                // num de paràmetres AU
FLOAT* pAU = NULL;            // paràmetres AU
FLOAT scale;               // escala
FLOAT rotationXYZ[3];      // paràmetres rotació (x,y,z)
FLOAT translationXYZ[3];   // paràmetres traslació (x,y,z)
UINT pointcloud;           // nombre punts plens del vector 2D
FT_VECTOR2D* pPts2D;          // vector 2D
```

Estructura del fitxer binari.

Ampliació de *FaceTrackingVisualization* . Funcions implementades

Tal com s'ha detallat amb anterioritat, s'han hagut d'implementar una sèrie de funcions de tractament de les imatges per a capturar la informació necessària pel projecte:

Segmentació del fons de la imatge

Per tal de separar la cara de l'objecte de la resta de la imatge, s'han aplicat una sèrie de funcions que ens permetran l'obtenció d'imatges més petites, amb la informació essencial pel reconeixement. En primer lloc, s'ha segmentat el fons de la cara del subjecte, fent un recorregut per tots els punts de la imatge i calculant si cadascun d'ells correspon o no a la cara del subjecte. Podem veure el codi a continuació.

```
    // Segmentem fons
    //***************
    int x = 0;
    int y = 0;
    for (x=(img2->width-1);x>=0;x=x--)
        for (y=(img2->height-1);y>=0;y--)
            if(!comparePixel(rectFace,x,y))
                paintPixel(img2,x,y,C_RGB_BLACK);
                       // si el pixel no és dins del rectangle cara, pinta'l de negre
```

Segmentació del fons.

La funció *"comparePixel()"* ens dona informació sobre si el píxel passat pertany o no al rectangle del subjecte. En cas negatiu, el píxel es converteix a color negre amb la funció "*paintPixel()*", per tal de segmentar-lo de la resta de la imatge.

En el codi següent es detalla la implementació d'aquestes funcions.

```
//*****************************************
// Funcions per detecció i modificació del fons
//*****************************************
void paintPixel(IplImage* in,int x, int y,CvScalar color) throw()
{   // comprova que els punt es dins la imatge i el modifica amb el color passat.
    if(in != NULL)
        {
            if(y>=0 && y<(in->height) && x>=0 && x<(in->width))
            {   // modifica el pixel a nivel de byte sobre el vector de la imatge.
                char *data = in->imageData;
                data[y*in->widthStep+x*in->nChannels]=color.val[0];
                data[y*in->widthStep+x*in->nChannels+1]=color.val[1];
                data[y*in->widthStep+x*in->nChannels+2]=color.val[2];
            }
        }
    }
     bool comparePixel(RECT face,int x, int y) throw()
{   // comprova si el punt passat és dins del rectangle
    if (x>= face.left && x<= face.right ){   // retorna TRUE si és dins, false si no
        if (y>=face.top && y<=face.bottom){
            return TRUE;}
        else
            return false;
    }
    else
        return false;
}
```

Detecció i modificació del fons.

A més, s'ha incorporat una nova funció que ens permet gravar l'estructura amb les característiques facials abans comentada, en un fitxer binari per a la seva utilització posterior. La funció utilitzada per a la gravació del vector de característiques és la següent:

```
//*********************************************
//  Gravació del vector de característiques a disk
//*********************************************
Void saveFeaturesVector(LPCTSTR Gridname)
{
            //LPCTSTR GridName = TEXT("img/%s_grid.bin");
            HRESULT  mallagrid  =  StringCchPrintfW(screenshotName,  _countof(screenshotName),
GridName, timeString);

            ofstream out(screenshotName, ios::binary);
            /*
            //Estructura del vector a gravar
            UINT numAU;
            FLOAT* pAU = NULL;
            FLOAT scale;
            FLOAT rotationXYZ[3];
            FLOAT translationXYZ[3];
            UINT pointcloud;
            FT_VECTOR2D* pPts2D;
            */
            out.write((char *) &numAU, sizeof(UINT));  // escrivim  valors num AU

            for(int i = 0; i< numAU; i++){
                float f = (float) pAU[i];
                out.write((char *) &f, sizeof(float));  // escrivim  valors pAU
              }

            out.write((char *) &scale, sizeof(float));  // escrivim  scale

            for(int i = 0; i< 3; i++){
                float f = (float) rotationXYZ[i];
                out.write((char *) &f, sizeof(float));  // escrivim  valors rotationXYZ
              }

            for(int i = 0; i< 3; i++){
                float f = (float) translationXYZ[i];
                out.write((char *) &f, sizeof(float));  // escrivim  valors translationXYZ
              }

            // escrivim  nombre de punts plens  del vector
            out.write((char *) &pointcloud, sizeof(UINT));
            // escrivim  valors del vector2D de vertex. El maxim definit són 121 punts
            for(int i = 0; i< 121; i++){
                FT_VECTOR2D v = pPts2D[i];
                float fx = v.x;
                float fy = v.y;
                out.write((char *) &fx, sizeof(float));  // escrivim  vector2D, valor x
                out.write((char *) &fy, sizeof(float));  // escrivim  vector2D, valor y
              }

            out.close();
}
```

Gravació del vector de característiques en fitxer binari.

Resultats obtinguts

A les dues imatges inferiors es pot observar la imatge inicial capturada pel dispositiu Kinect (figura 18) i la resultant del tractament aplicat, retallant la imatge fins adaptar-la al quadrat delimitat pel perímetre de la malla (figura 19).

Figura 18: Imatge del subjecte obtinguda en RGB i resolució 640x480.

Figura 19: Imatge final obtinguda, utilitzada per a la inclusió a la BBDD.

6.3.4. Creació de la BBDD d'imatges

Partint del desenvolupament anterior, podrem crear una primera base de dades d'imatges amb informació RGB/Depth que servirà de base per les diferents proves en alguns dels prototipus de reconeixement facial.
A nivell experimental, s'ha escollit un model de base de dades basada en un repositori de fitxers, amb un fitxer índex que es farà servir per a la seva càrrega a la solució de reconeixement.

Arquitectura de la base de dades

La base de dades estarà formada per una carpeta de nom "bbdd", on es dipositaran las diferents vistes dels subjectes (només vistes frontals de les cares) i les dades 2D i 3D. El nom dels fitxers tindrà el format: "xxxxx_F.jpg" (frontal), "xxxxx_L.jpp2 (vista esquerra), "xxxxx_R.jpg" (vista dreta) i "xxxxx_grid.bin2 (pel fitxer amb els vectors 2D i 3D).

L'índex de la bbdd, el formarà un fitxer "csv", amb valors separats per ";" en el que s'inclourà una línia per a cada subjecte amb els paràmetres següents:

xxxxx_F.jpg; xxxxx_F_grid.bin;id

sent la primera imatge del fitxer (amb identificador 0), la imatge de test (xxxx_X.jpg), id, un enter identificador del subjecte (*label*) i la resta d'imatges, les de la bbdd d'entrenament.

Un exemple de fitxer amb 1 subjecte de test i 5 imatges de bbdd (imatges.csv) serà el següent:

```
xxxxx.jpg;xxxxx_grid.bin;0
aaaaa_F.jpg;aaaaa_F_grid.bin;1
aaaaa_L.jpg;aaaaa_L_grid.bin;1
aaaaa_R.jpg;aaaaa_R_grid.bin;1
bbbbb_F.jpg;bbbbb_F_grid.bin;2
bbbbb_L.jpg;bbbbb_R_grid.bin;2
bbbbb_R.jpg;bbbbb_L_grid.bin;2
ccccc_F.jpg;ccccc_F_grid.bin;3
ccccc_L.jpg;ccccc_R_grid.bin;3
ccccc_R.jpg;ccccc_L_grid.bin;3
ddddd_F.jpg;ddddd_F_grid.bin;4
ddddd_L.jpg;ddddd_R_grid.bin;4
ddddd_R.jpg;ddddd_L_grid.bin;4
eeeee_F.jpg;eeeee_F_grid.bin;5
eeeee_L.jpg;eeeee_R_grid.bin;5
eeeee_R.jpg;eeeee_L_grid.bin;5
fffff_F.jpg;fffff_F_grid.bin;6
fffff_L.jpg;fffff_R_grid.bin;6
fffff_R.jpg;fffff_L_grid.bin;6
```

Imatges.csv.

Per la captura de les imatges s'ha fet servir un entorn controlat, amb il·luminació difusa. Totes les imatges han estat capturades des d'una distància entre 0,8 y 1,5 m del dispositiu Kinect, s'han escalat per tal que les cares tinguin una superfície similar, i s'ha retallat el primer pla a unes mides de 200x200 píxels.

Càrrega de les dades

Per a la càrrega de les imatges i els identificadors, utilitzarem la següent funció mitjançant la qual llegirem les imatges del repositori i omplirem el vector d'objectes tipus OpenCv Mat anomenat "imatges", associat a un vector d'identificadors de subjecte "labels". A més, també s'ha previst la utilització d'una funció auxiliar "carregaVector2D()", que recuperarà la informació del fitxer que conté les dades de la malla i les carregarà a un vector d'estructures 2D.

```
vector<Mat> images;
vector<int> labels;

//*****************************************
// lectura index bbdd CSV i càrrega de vectors
//*****************************************
static void read_csv(const string& filename, vector<Mat>& images, vector<int>& labels, char
separator = ';') {
    std::ifstream file(filename.c_str(), ifstream::in);
    if (!file) {
        string error_message = " Input file not valid. Please check Filename.";
        CV_Error(CV_StsBadArg, error_message);
    }
```

```
    string line, pathF, , pathL, , pathR, , pathBin ,classlabel;
    while (getline(file, line)) {

        stringstream liness(line);
        getline(liness, pathF, separator);
        getline(liness, pathL, separator);
        getline(liness, pathR, separator);
        getline(liness, pathBin, separator);
        getline(liness, classlabel);

        if(!path.empty() && !classlabel.empty()) {

            images.push_back(imread(pathF, 0));
            images.push_back(imread(pathL, 0));
            images.push_back(imread(pathR, 0));
             carregaVector2D(pathBin)// funció que carrega estructura vectors 2D i 3D
            labels.push_back(atoi(classlabel.c_str()));
        }
    }
}
```

Càrrega de les imatges.

6.3.5. Reconeixement facial. Algorisme PCA (Eigenfaces)

Com hem comentat anteriorment, el reconeixement facial es plantejat per l'algorisme PCA, com a un problema de reconeixement en dues dimensions. Parteix de la base que les fotografies de les cares són normalment frontals, pel que les seves característiques poden ser descrites per un conjunt de vistes en 2D.

Amb PCA, la imatge a analitzar i la galeria d'imatges han de ser de la mateixa grandària i han de ser normalitzades prèviament per alinear els ulls i boques dels subjectes a les imatges.

L'algorisme processarà d'imatges amb l'objectiu d'aconseguir reconeixement facial mitjançant l'ús de la descomposició en valors principals (PCA) i Eigenfaces, tot seguint els següents passos:

1. Emmagatzemarà un conjunt d'imatges d'entrenament de diferents persones, mirant de tenir subconjunts d'imatges per a cada persona que continguin diferents postures, condicions d'il·luminació, etc.

2. Crearà una matriu formada per la nova imatge d'entrada i les ja emmagatzemades en la base de dades. Mitjançant el procés a dalt descrit es calculen els *Eigenvectors* a partir de la matriu de covariància.

3. Una vegada obtinguts els vectors característics, comparará les distàncies entre el vector que representa a la imatge original amb la resta.

4. Establirà un llindar de discerniment (*confidence*). Si el menor valor del pas anterior és menor que aquest llindar, la imatge de la cara d'entrada serà considerada com a coneguda. Pel contrari si és major, serà considera desconeguda.

Veurem que els valors de les distàncies depenen en certa mesura de la grandària de la base de dades, ja que la matriu de covariància i els vectors característics són calculats a partir de la matriu formada per la imatge d'entrada i les ja emmagatzemades, per la qual cosa el llindar de discerniment hauria de ser dinàmic i

adaptar-se segons la variació de les distàncies entre cares, o el que és el mateix, segons la grandària de la base de dades que emmagatzema les imatges de les cares.

Implementació de PCA i Eigenfaces en OpenCV

La forma més simple de fer algorismes de reconeixement facial passa per fer servir les classes que implementa OpenCV pel reconeixement PCA. Aquest algorisme és ràpid i necessita pocs recursos de memòria ja que bàsicament treballa amb reducció dimensional. Com a base de partida, s'han fet servir els treballs de Philip Wagner detallats al seu document "Guide to Face recognition with OpenCV2" [22].

Malgrat tot, els algorismes PCA, són sensibles a la translació de les imatges tractades (el que vol dir que si les imatges no estan ben alineades, serà difícil el reconèixer-les). També es veuen afectats pels escalats pel que les imatges han de tenir la mateixa escala per que l'algorisme funcioni. Per últim, els fons de les imatges han de ser uniformes. L'algorisme també treballa millor si els fons de les diferents imatges són similars o fins i tot si no existeix.

Per totes aquestes raons, abans de fer servir un algorisme PCA per reconeixement, cal establir algunes fases de preprocessament de les imatges. Aquestes poden ser:

- Detecció de cara i extracció: Detectar la cara fent servir "OpenCV Face Detection", per tal de retallar de la imatge original un primer pla del subjecte, abans de reconèixer la cara.
- Normalització d'il·luminació: és important d'aplicar algun mètode de normalització, com ara equalització d'histograma o utilitzar altres mètodes.
- Escalat: Cal assegurar-se que les imatges estan totes escalades a la mateixa grandària i que la superfície presentada per cadascuna de les cares a les imatges és similar.

Base de dades d'imatges utilitzada pel reconeixement PCA

Per a la implementació d'aquest prototipus, utilitzarem una base de dades diferent a l'especificada en el punt anterior. La nova base de dades estarà formada, igualment, per un repositori (carpeta "input") on es dipositaran las vistes del diferents subjectes (només primers plans frontals de la cara) en format RGB. El nom dels fitxers tindrà ara en canvi el format: "id-n.jpg" , on id serà un identificador únic del subjecte i n en nombre d'ordre de la imatge . L'índex de la base de dades, el formarà un fitxer "csv", (valors separats per ";") en el que s'inclourà una línia per a cada subjecte amb els paràmetres següents: " id-x.jpg;id".

La darrera imatge del fitxer índex serà la imatge de test (**1-x.jpg**) i la resta d'imatges, les de la base de dades, agrupades però per subjecte. Per la captura de les imatges s'ha fet servir igualment un entorn controlat, amb il·luminació difusa, una distància focal entre 0,8 y 1,5 m del dispositiu Kinect, i a més s'ha fet un preprocessament consistent en l'escalat i el retallat del primer pla del subjecte a la mida de 200x200 píxels. L'índex de la base de dades de test (input.csv) és el següent:

```
input/0-1.jpg;0
input/1-1.jpg;1
input/1-2.jpg;1
input/1-3.jpg;1
input/2-1.jpg;2
input/2-2.jpg;2
input/3-1.jpg;3
input/4-1.jpg;4
input/4-2.jpg;4
input/4-3.jpg;4
input/5-1.jpg;5
input/6-1.jpg;6
input/7-1.jpg;7
input/8-1.jpg;8
input/9-1.jpg;9
input/10-1.jpg;10
input/10-2.jpg;10
input/10-3.jpg;10
input/10-1.jpg;10
input/11-1.jpg;11
input/11-2.jpg;11
input/11-3.jpg;11
input/12-1.jpg;12
input/1-x.jpg;1
```

Input.csv.

A la imatge inferior, podem veure un exemple de la base de dades d'imatges utilitzada, representada per un repositori bàsic amb vint-i-tres imatges d'entrenament i quatre de test.

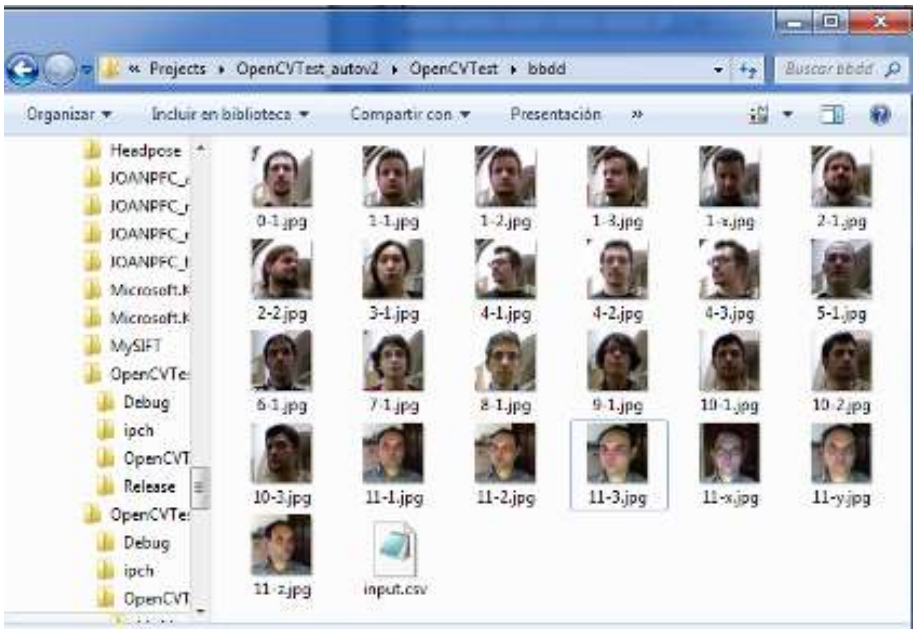


Figura 20: Repositori d'imatges de la base de dades amb l'índex.

Implementació del prototipus de detecció PCA.

Per tal de crear el model Eigenface i entrenar-lo amb les imatges d'entrenament, crearem un objecte *FaceRecognizer* i l'alimentarem amb el vector d'imatges d'entrenament (prèviament, hem extret la imatge de test del vector). Després

passarem la imatge de test i obtindrem la predicció de resultat. Per últim, gravarem o presentarem per pantalla els resultats obtinguts del reconeixement.

La normalització de les imatges la realitzarem utilitzant la funció "*norm_0_255()*" descrita a continuació, que converteix les imatges a escala de grisos.

```
//**********************************
// Normalització d'imatges
//**********************************
static Mat norm_0_255(InputArray _src) {
    Mat src = _src.getMat();
    // Crear i retornar imatge normalizada
    Mat dst;
    switch(src.channels()) {
    case 1:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC1);
        break;
    case 3:
        cv::normalize(_src, dst, 0, 255, NORM_MINMAX, CV_8UC3);
        break;
    default:
        src.copyTo(dst);
        break;
    }
    return dst;
}
```

Normalització d'imatges.

La següent funció, "*read_csv()*", llegirà les imatges del repositori i les inclourà en un vector d'imatges (images) pel seu tractament. A més a més, crearà un vector índex (labels), amb el identificador associat a cada imatge. La darrera posició del vector, inclourà la imatge de test.

```
//**************************************
//  Lectura CSV i càrrega vectors
//**************************************

static void read_csv(const string& filename, vector<Mat>& images, vector<int>& labels, char
separator = ';') {

    std::ifstream file(filename.c_str(), ifstream::in);
    if (!file) {
        string error_message = "No valid input file was given, please check the given
filename.";
        CV_Error(CV_StsBadArg, error_message);
    }

    string line, path, classlabel;
    while (getline(file, line)) {
        stringstream liness(line);
        getline(liness, path, separator);
        getline(liness, classlabel);
        if(!path.empty() && !classlabel.empty()) {
            images.push_back(imread(path, 0));
            labels.push_back(atoi(classlabel.c_str()));
        }
    }
}
```

Lectura fitxers csv.

La funció "*predict*()" d'OpenCV utilitza una implementació de SVM (Support Vertor Machines) per a la categorització. Calcula la imatge més adient a la passada com a objecte Mat de referència i retorna el valor numèric de la etiqueta corresponent a la imatge de predicció del vector d'entrenament i un valor "*confidence*" de fiabilitat de la predicció. La seva implementació es pot veure al fragment de codi següent:

```
    // creació objecte FaceRecognizer
    Ptr<FaceRecognizer> model = createEigenFaceRecognizer();

    // Entrenament amb vector imatges
    model->train(images, labels);

    // predicció a partir de la imatge de test
    int predictedLabel = model->predict(testSample);
    string result_message = format("Predicted class = %d / Actual class = %d.", predictedLabel,
testLabel);
    cout << result_message << endl;

    // recuperem els eigenvalues i els guardem en una matriu
    Mat eigenvalues = model->getMat("eigenvalues");

    // fem el mateis amb els EigenVectors
    Mat W = model->getMat("eigenvectors");

    // Recuperem la imatge promig
    Mat mean = model->getMat("mean");
```

Implementació predicció PCA.

A la imatge següent, podem veure un exemple de la imatge promig (esquerra) i dels Eigenfaces creats en el funcionament del prototipus.

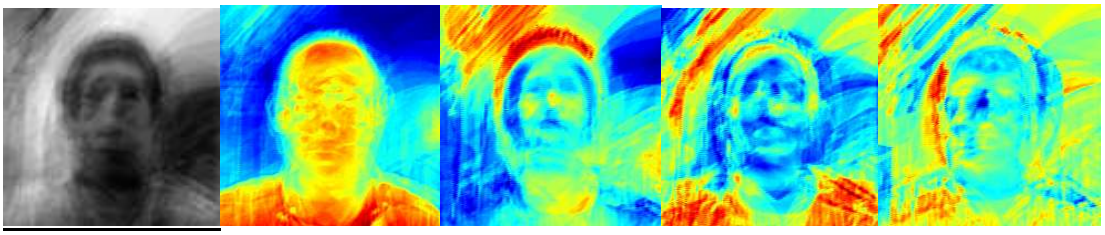


Figura 21: Imatge promig i exemple dels eigenfaces obtinguts.

Resultats obtinguts

A la imatge inferior (figura 22) podem veure la imatge de test utilitzada (esquerra), mentre que a la dreta podem observar la imatge de l'usuari, escollida del conjunt d'entrenament, que l'algorisme ha predit entre una col·lecció de 23 imatges de subjectes diferents. L'índex de fiabilitat ha estat de 7972,20.

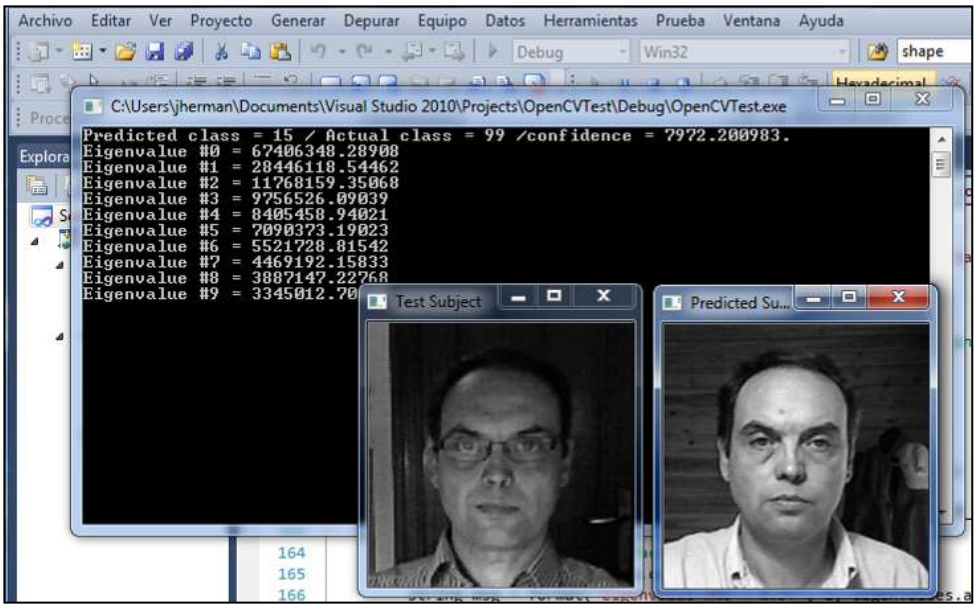


Figura 22: Presentació dels resultats obtinguts.

Identificació en temps real

Una millora de la implementació anterior consisteix en la identificació de subjectes en temps real. Mentre que en el prototipus descrit anteriorment la detecció es fa mitjançant un conjunt d'entrenament i una imatge de test especificats al fitxer d'índex, en aquest nou prototipus la imatge de test es captura en temps real mitjançant el dispositiu Kinect.

Per tal de poder fer una identificació el més precisa possible, abans de passar la imatge capturada al sistema d'identificació, aplicarem detecció de la cara del subjecte en temps real, retallarem un primer pla dels subjecte i l'escalarem a unes mides de 200x200 punts per fer les imatges compatibles amb les emmagatzemades a la base de dades. Per últim, farem una conversió a escala de grisos.

La funció "*detectFaces*()" que permetrà la detecció de la cara en temps real és la següent:

```
void detectFaces( IplImage *img )
{
    int i;

    CvSeq *faces = cvHaarDetectObjects(
            img,
            cascade,
            storage,
            1.1,
            3,
            0 /*CV_HAAR_DO_CANNY_PRUNNING*/,
            cvSize( 40, 40 ) );

    for( i = 0 ; i < ( faces ? faces->total : 0 ) ; i++ ) {
        r = ( CvRect* )cvGetSeqElem( faces, i );
              // Origin point of detected face
        cvRectangle( img,
            cvPoint( r->x, r->y ),
            cvPoint( r->x + r->width, r->y + r->height ),
            CV_RGB( 255, 0, 0 ), 1, 8, 0 ); // draw rectangle over face
```

```
        // save the center of the face
        centerX = r->x - (r->width)/2;
        centerY = r->y - (r->height)/2;
    }

    cvShowImage( "video", img );
}
```

Detecció en temps real.

La funció principal, carregarà els fitxers d'imatge de la base de dades d'entrenament descrits a index.csv, aplicará una equalització i instanciarà l'entrenament del objecte *FaceRecognizer*.

Una vegada finalitzat l'entrenament, començarà una captura periòdica d'imatges del flux de vídeo, instanciant després de la captura el reconeixement de la imatge, prèviament tractada i equalitzada.

El detall d'aquesta funció principal es pot veure a continuació:

```
int main( int argc, char** argv )
{
    CvCapture* capture = cvCaptureFromCAM( CV_CAP_ANY );
    IplImage* frame = cvCreateImage( cvSize(640,480), IPL_DEPTH_8U, 3 );

    int       key = 0;
    double counter = 0;
    double intpart;
    char      *filename = "haarcascade_frontalface_alt.xml";
    cascade = ( CvHaarClassifierCascade* )cvLoad( filename, 0, 0, 0 );
    storage = cvCreateMemStorage( 0 );

    read_and_train();   // carreguem set d'entrenament

    capture = cvCaptureFromCAM( 0 );
    assert( cascade && storage && capture );
    cvNamedWindow( "video", 1 );

    while( key != 27 ) {  // If ESC is not pressed
        frame = cvQueryFrame( capture );
        if( !frame ) {
            fprintf( stderr, "Cannot query frame!\n" );
            //break;
        }
        frame->origin = 0;
        detectFaces( frame );
        if(modf(counter/3,&intpart)==0){
        // Auto mode: Make an automatic reco every 3 times
        //if (key == 'r') {
        // Manual mode: by pressing 'r', capture image and start reconaissance.
            Mat mtx(frame);  // working copy of the  test image
            //call recognize with normalized image mat
            recognize(  crop_and_norm(mtx));
        }
        counter++;
        key = cvWaitKey( 10 );
    }
    cvReleaseCapture( &capture );
    cvDestroyWindow( "video" );
    cvDestroyWindow( "Predicted Subject" );
    cvReleaseHaarClassifierCascade( &cascade );
    cvReleaseMemStorage( &storage );
    return 0;
}
```

Funció principal del prototipus de reconeixement en temps real.

Els resultats obtinguts es poden apreciar en la figura següent (figura 23), on a la imatge petita es pot veure la imatge del conjunt d'entrenament predita per l'algorisme.

En aquest cas la predicció ha estat encertada i l'índex de fiabilitat elevat: 10811.

Figura 23: Presentació dels resultats obtinguts per l'algorisme de reconeixement en temps real.

7. Disseny i implementació

Per a la implementació del prototipus del projecte partirem del diagrama de blocs principal especificat a la figura 11. Per al seu desenvolupament ens basarem en el codi emprat al reconeixement PCA, que com s'ha comentat anteriorment, utilitza SVM (Support Vector Machines) per a la categorització de les imatges.

El diagrama de blocs específic de l'algoritme implementat en el prototipus és el següent:

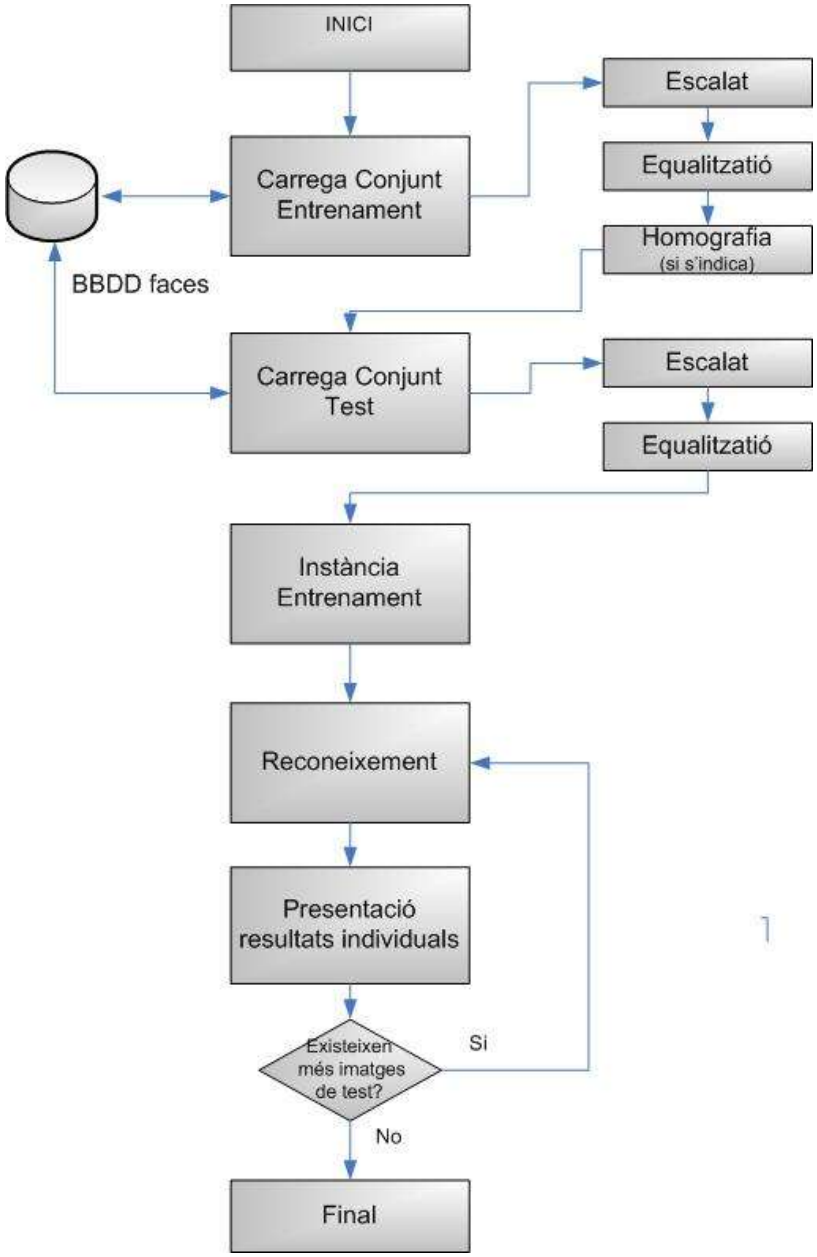


Figura 24: Diagrama de blocs del prototipus del projecte.

En aquest cas, introduirem algunes variacions de tractament d'imatge amb l'objectiu final de millorar la fiabilitat en el reconeixement. Les imatges de test, es carregaran també un vector d'imatges amb un tractament previ d'escalat i equalització.

Respecte a la base de dades d'entrenament, per a cada imatge carregada d'un subjecte s'extrauran les característiques principals i s'aplicarà Homografia per tal de calcular la matriu de transformació respecte a la imatge principal frontal del subjecte (id-0.jpg).

Una vegada calculada aquesta matriu, es farà una transformació de la imatge (warp) d'entrenament i es guardarà al vector d'imatges d'entrenament. El detall de funcionament dels algorismes de detecció de característiques, correspondència i càlcul de la matriu de transformació es detallen en el punt 7.1.3 i 7.1.4 d'aquest document.

El resultat serà l'obtenció d'un vector d'imatges transformades, que s'utilitzarà per l'entrenament previ al reconeixement PCA. Una vegada efectuat aquest entrenament, s'instanciarà el reconeixement per cadascuna de les imatges dels subjectes de test, obtenint una matriu de resultats.

Es repetirà després el procés, aquesta vegada creant el vector d'imatges d'entrenament sense transformar les imatges de test (sense aplicar Homografia), només amb l'aplicació prèvia d'escalat de les imatges i d'equalització. S'instanciarà de nou el procés de reconeixement de les imatges de test i s'obtindrà una segona matriu de resultats que podem comparar amb la obtinguda anteriorment.

Per tal de permetre un canvi fàcil entre ambdós tipus de proves, la funció principal de l'algoritme permetrà l'activar o desactivar la transformació per Homografia de les imatges d'entrenament abans de la seva càrrega al vector corresponent.

7.1. Fase d'entrenament

La fase d'entrenament comporta la càrrega de les imatges del repositori indicades als fitxers d'índex en vectors d'imatges (un per l'entrenament i l'altre pel test) i la crida a la instancia de l'entrenament en sí mateix. Com s'ha detallat abans ,existeix una versió de l'entrenament que utilitzarà imatges transformades utilitzant Homografia i un altre versió que no la utilitzarà.

7.1.1. Base de dades d'imatges

Per a comprovar el funcionament del prototipus, utilitzarem una base de dades similar a la utilitzada en el prototipus de reconeixent PCA, especificada a l'apartat 6.3.5, complementada amb imatges procedents de la base de dades d'imatges BIO ID [2] fins a un total de 200 imatges d'entrenament i de 50 imatges de test, corresponents a 10 subjectes diferents. Les imatges han estat prèviament tractades (equalització),

[2] La base de dades BIO ID està disponible pel seu ús no comercial a la pàgina de descàrrega de la companyia: http://support.bioid.com/downloads/software/bioid-face-database.html

retallades *a l'àrea delimitada per la malla* corresponent a la cara del subjecte, i escalades fins a una grandària de 100x100 píxels.

Un exemple de les imatges utilitzades al conjunt d'entrenament de la base de dades es pot veure a la figura 25 a continuació.

Figura 25: Mostra exemple de les imatges de la base de dades utilitzada.

Com a detall principal a tenir en compte, l'algorisme de tractament aplicat, necessitarà identificar una imatge facial frontal i neutre de cada subjecte de test com a imatge principal del seu subconjunt de test. Aquesta imatge principal s'identificarà com a "id-0.jpg", sent "id " l'identificador del subjecte. La imatge es col·locarà al fitxer d'índex "input.csv" en la primera posició de les imatges pròpies de cada subjecte. La resta d'imatges d'un mateix subjecte es detallaran a continuació.

Podem veure a continuació un exemple del contingut del fitxer "input.csv" utilitzat al projecte, que conté un total de 100 imatges d'entrenament, i on s'han ressaltat en vermell les imatges principals de cada subjecte:

input/1-0.jpg;1	input/4-5.jpg;4	input/8-0.jpg;8	input/10-5.jpg;10
input/1-1.jpg;1	input/4-6.jpg;4	input/8-1.jpg;8	input/10-6.jpg;10
input/1-2.jpg;1	input/4-7.jpg;4	input/8-2.jpg;8	input/10-7.jpg;10
input/1-3.jpg;1	input/4-8.jpg;4	input/8-3.jpg;8	input/10-8.jpg;10
input/1-4.jpg;1	input/4-9.jpg;4	input/8-4.jpg;8	input/10-9.jpg;10
input/1-5.jpg;1	input/5-0.jpg;5	input/8-5.jpg;8	input/11-0.jpg;11
input/1-6.jpg;1	input/5-1.jpg;5	input/8-6.jpg;8	input/11-1.jpg;11
input/1-7.jpg;1	input/5-2.jpg;5	input/8-7.jpg;8	input/11-2.jpg;11
input/1-8.jpg;1	input/5-3.jpg;5	input/8-8.jpg;8	input/11-3.jpg;11
input/1-9.jpg;1	input/5-4.jpg;5	input/8-9.jpg;8	input/11-4.jpg;11

input/3-0.jpg;3	input/5-5.jpg;5	input/9-0.jpg;9	input/11-5.jpg;11
input/3-1.jpg;3	input/5-6.jpg;5	input/9-1.jpg;9	input/11-6.jpg;11
input/3-2.jpg;3	input/5-7.jpg;5	input/9-2.jpg;9	input/11-7.jpg;11
input/3-3.jpg;3	input/5-8.jpg;5	input/9-3.jpg;9	input/11-8.jpg;11
input/3-4.jpg;3	input/5-9.jpg;5	input/9-4.jpg;9	input/11-9.jpg;11
input/3-5.jpg;3	input/6-0.jpg;6	input/9-5.jpg;9	input/12-0.jpg;12
input/3-6.jpg;3	input/6-1.jpg;6	input/9-6.jpg;9	input/12-1.jpg;12
input/3-7.jpg;3	input/6-2.jpg;6	input/9-7.jpg;9	input/12-2.jpg;12
input/3-8.jpg;3	input/6-3.jpg;6	input/9-8.jpg;9	input/12-3.jpg;12
input/3-9.jpg;3	input/6-4.jpg;6	input/9-9.jpg;9	input/12-4.jpg;12
input/4-0.jpg;4	input/6-5.jpg;6	input/10-0.jpg;10	input/12-5.jpg;12
input/4-1.jpg;3	input/6-6.jpg;6	input/10-1.jpg;10	input/12-6.jpg;12
input/4-2.jpg;4	input/6-7.jpg;6	input/10-2.jpg;10	input/12-7.jpg;12
input/4-3.jpg;4	input/6-8.jpg;6	input/10-3.jpg;10	input/12-8.jpg;12
input/4-4.jpg;4	input/6-9.jpg;6	input/10-4.jpg;10	input/12-9.jpg;12

Input.csv.

Les imatges de test que utilitzarà l'algoritme es carregaran al vector de test a partir del contingut del fitxer "InputTest.csv". S'utilitzaran un total de 50 imatges, (5 diferents i no utilitzades al conjunt d'entrenament per a cada subjecte) per tal de comprovar el funcionament del prototipus. El detall del fitxer de test és el següent:

input/1-x.jpg;1	input/5-z.jpg;5	input/8-n.jpg;8
input/1-y.jpg;1	input/5-n.jpg;5	input/8-h.jpg;8
input/1-z.jpg;1	input/5-h.jpg;5	input/10-y.jpg;10
input/1-n.jpg;1	input/6-x.jpg;6	input/10-z.jpg;10
input/1-h.jpg;1	input/6-y.jpg;6	input/10-n.jpg;10
input/3-x.jpg;3	input/6-z.jpg;6	input/10-h.jpg;10
input/3-y.jpg;3	input/6-n.jpg;6	input/11-x.jpg;11
input/3-z.jpg;3	input/9-x.jpg;9	input/11-y.jpg;11
input/3-n.jpg;3	input/9-y.jpg;9	input/11-z.jpg;11
input/3-h.jpg;3	input/9-z.jpg;9	input/11-n.jpg;11
input/4-x.jpg;4	input/9-n.jpg;9	input/11-h.jpg;11
input/4-y.jpg;4	input/9-h.jpg;9	input/12-x.jpg;12
input/4-z.jpg;4	input/10-x.jpg;10	input/12-y.jpg;12
input/4-n.jpg;4	input/6-h.jpg;6	input/12-z.jpg;12
input/4-h.jpg;4	input/8-x.jpg;8	input/12-n.jpg;12
input/5-x.jpg;5	input/8-y.jpg;8	input/12-h.jpg;12
input/5-y.jpg;5	input/8-z.jpg;8	

InputTest.csv.

A continuació (figura 26) es pot veure un subconjunt de les 50 imatges prèviament equalitzades utilitzades al conjunt de test del projecte. Aquestes imatges no han estat utilitzades a l'entrenament, per tal de no esbiaixar els resultats de la prova.

Figura 26: Mostra de les imatges utilitzades al conjunt de test.

7.1.2. Escalat d'imatges i equalització

Pel tractament de les imatges s'ha utilitzat funcions de retallat i escalat que respecten les proporcions originals de la imatge, descrita a continuació. La funció "resize()" [3], crea una nova imatge a partir de la original, amb la nova grandària desitjada, respectant les proporcions i retallant les parts no desitjades de manera que només es mostraran els píxels de la imatge original, evitant l'afegir espais addicionals en blanc.

```
IplImage* resizeImage(const IplImage *origImg, int newWidth,
 int newHeight, bool keepAspectRatio)
{
 IplImage *outImg = 0;
 int origWidth;
 int origHeight;
 if (origImg) {
     origWidth = origImg->width;
     origHeight = origImg->height;
 }
 if (newWidth <= 0 || newHeight <= 0 || origImg == 0
     || origWidth <= 0 || origHeight <= 0) {
     //cerr << "ERROR: Bad desired image size of " << newWidth
     //  << "x" << newHeight << " in resizeImage().\n";
     exit(1);
 }
```

[3] Les rutines de transformació d'imatges s'han basat en els treballs de Servin Emami, descrits a http://www.shervinemami.info/imageTransforms.html.

```
if (keepAspectRatio) {
    // Resize the image without changing its aspect ratio,
    // by cropping off the edges and enlarging the middle section.
    CvRect r;
    // input aspect ratio
    float origAspect = (origWidth / (float)origHeight);
    // output aspect ratio
    float newAspect = (newWidth / (float)newHeight);
    // crop width to be origHeight * newAspect
    if (origAspect > newAspect) {
        int tw = (origHeight * newWidth) / newHeight;
        r = cvRect((origWidth - tw)/2, 0, tw, origHeight);
    }
    else {  // crop height to be origWidth / newAspect
        int th = (origWidth * newHeight) / newWidth;
        r = cvRect(0, (origHeight - th)/2, origWidth, th);
    }
    IplImage *croppedImg = cropImage(origImg, r);

    // Call this function again, with the new aspect ratio image.
    // Will do a scaled image resize with the correct aspect ratio.
    outImg = resizeImage(croppedImg, newWidth, newHeight, false);
    cvReleaseImage( &croppedImg );
}
else {

    // Scale the image to the new dimensions,
    // even if the aspect ratio will be changed.
    outImg = cvCreateImage(cvSize(newWidth, newHeight),
        origImg->depth, origImg->nChannels);
    if (newWidth > origImg->width && newHeight > origImg->height) {
        // Make the image larger
        cvResetImageROI((IplImage*)origImg);
        // CV_INTER_LINEAR: good at enlarging.

        // CV_INTER_CUBIC: good at enlarging.
        cvResize(origImg, outImg, CV_INTER_LINEAR);
    }
    else {
        // Make the image smaller
        cvResetImageROI((IplImage*)origImg);
        // CV_INTER_AREA: good at shrinking (decimation) only.
        cvResize(origImg, outImg, CV_INTER_AREA);
    }

}
return outImg;
}
```

Escalat proporcional d'imatges.

Aquesta funció utilitza com a secundària la funció "*cropImage()*" també detallada a continuació.

```
IplImage* cropImage(const IplImage *img, const CvRect region)
{
// Returns a new image that is a cropped version (rectangular cut-out)
// of the original image.

    IplImage *imageCropped;
    CvSize size;

    if (img->width <= 0 || img->height <= 0
        || region.width <= 0 || region.height <= 0) {
        //cerr << "ERROR in cropImage(): invalid dimensions." << endl;
        exit(1);
    }
    if (img->depth != IPL_DEPTH_8U) {
        //cerr << "ERROR in cropImage(): image depth is not 8." << endl;
        exit(1);
    }
    // Set the desired region of interest.
    cvSetImageROI((IplImage*)img, region);
```

```
    // Copy region of interest into a new iplImage and return it.
    size.width = region.width;
    size.height = region.height;
    imageCropped = cvCreateImage(size, IPL_DEPTH_8U, img->nChannels);
    cvCopy(img, imageCropped);   // Copy just the region.

    return imageCropped;
}
```

Funció CropImage.

Equalització

L'equalització és un mètode de modificació de l'histograma (representació gràfica del rang d'intensitat dels píxels de la imatge) per tal de millorar el contrast i ampliar rang dinàmic. En l'algoritme s'ha fet servir la funció "*equalizeHist()*" incorporada a les llibreries OpenCV.

En la figura 27 podem observar una imatge sense equalitzar i el seu histograma corresponent, on es pot veure que els tons mitjos són els predominants, mentre que el nombre de píxels en les zones d'intensitat baixa i alta és molt petit.

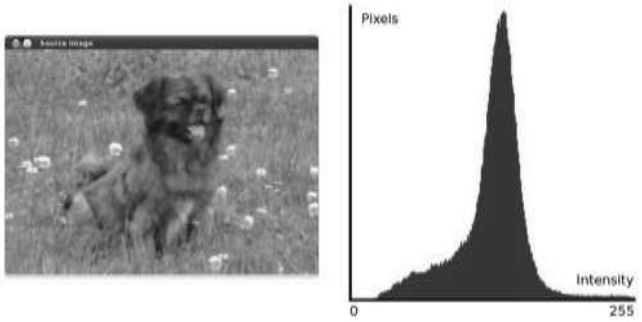


Figura 27: Imatge original sense equalitzar amb el seu histograma corresponent.

En la figura 28 podem veure com l'histograma de la imatge original ha ampliat el seu rang dinàmic. Ara els nivells d'intensitat estan més distribuïts al llarg de tota la corba de l'histograma. A la dreta, la imatge resultant on s'aprecia una millora significativa del contrast.

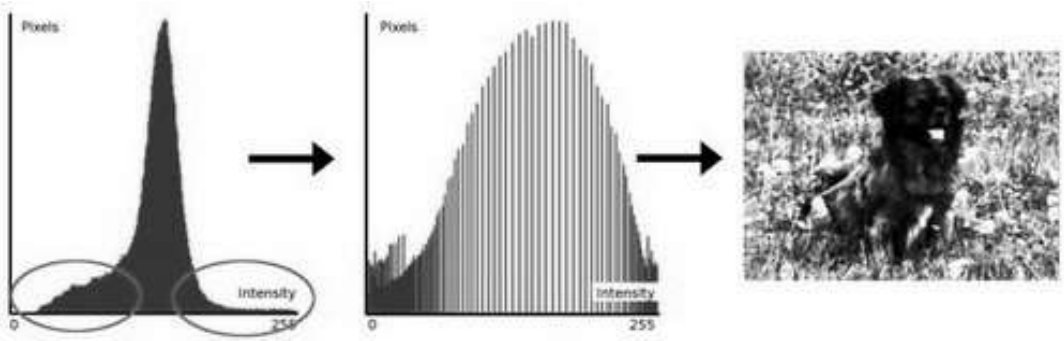


Figura 28: Evolució de l'histograma i imatge resultant.

7.1.3. Extracció de característiques

Per cadascuna de les imatges del conjunt d'entrenament aplicarem un procés de detecció de característiques. D'una banda, les extraurem de la imatge principal de cadascun dels subjectes, i d'altre banda de cadascuna de les imatges carregades per al subjecte en qüestió.

En primer lloc crearem un objecte *SurfFeatureDetector* i aplicarem el mètode *detection* per a les dues imatges (a l'exemple els objectes Opencv Mat anomenats img_object i img_scene), i crearem les seves dues matrius corresponents de descriptors (descriptors_object i descriptors_scene), que necessitarem després en la detecció de característiques coincidents a les dues imatges. Al fragment de codi següent es pot apreciar la implementació de la detecció dels punts de característiques de les imatges.

```
//-- Step 1: Detect the keypoints using SURF Detector
    int minHessian = 5;

    SurfFeatureDetector detector( minHessian );

    std::vector<KeyPoint> keypoints_object, keypoints_scene;

    detector.detect( img_object, keypoints_object );
    detector.detect( img_scene, keypoints_scene );

    //-- Step 2: Calculate descriptors (feature vectors)
    SurfDescriptorExtractor extractor;

    Mat descriptors_object, descriptors_scene;

    extractor.compute( img_object, keypoints_object, descriptors_object );
    extractor.compute( img_scene, keypoints_scene, descriptors_scene );
```

Detecció de punts de característiques.

Per tal de detectar les correspondències entre els punts de característiques de les imatges, utilitzarem una implementació de les llibreries FLANN, utilitzades normalment per a la cerca ràpida en espais d'alta dimensionalitat utilitzant tècniques de "*nearest neighbor*" [4]. La implementació d'aquesta llibreria es pot veure en el fragment de codi següent:

```
//-- Step 3: Matching descriptor vectors using FLANN matcher
    FlannBasedMatcher matcher;
    std::vector< DMatch > matches;
    matcher.match( descriptors_object, descriptors_scene, matches );
    double max_dist = 0; double min_dist = 100;

      //-- Quick calculation of max and min distances between keypoints
      for( int i = 0; i < descriptors_object.rows; i++ )
      { double dist = matches[i].distance;
        if( dist < min_dist ) min_dist = dist;
        if( dist > max_dist ) max_dist = dist;
      }
      printf("-- Max dist : %f \n", max_dist );
      printf("-- Min dist : %f \n", min_dist );
      //-- Draw only "good" matches (i.e. whose distance is less than 3*min_dist )
      std::vector< DMatch > good_matches;

      for( int i = 0; i < descriptors_object.rows; i++ )
```

[4] La informació sobre FLANN (Fast Library for Approximate Nearest Neighbors) es pot trobar a http://www.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN.

```
      //{ if( matches[i].distance < 3*min_dist )
      { if( matches[i].distance < 2*min_dist )
         { good_matches.push_back( matches[i]); }
      }

      Mat img_matches;
      drawMatches( img_object, keypoints_object, img_scene, keypoints_scene,
                   good_matches, img_matches, Scalar::all(-1), Scalar::all(-1),
                   vector<char>(), DrawMatchesFlags::NOT_DRAW_SINGLE_POINTS );
```

Detecció de correspondències entre característiques.

7.1.4. Homografia

A partir dels vectors de punts de característiques coincidents, calcularem mitjançant una Homografia la matriu de transformació (H) que ens permetrà adaptar la imatge de test a la imatge principal del subjecte. Una vegada obtinguda la matriu (H), farem una transformació (funció *warp*) de la imatge obtenint la imatge final transformada que guardarem al vector d'imatges d'entrenament.

La part de la solució que detecta les coincidències i transforma la imatge original es pot veure al fragment de codi adjunt, on s'implementa la funció "*findHomography()*" a tal efecte, fent servir l'algoritme RANSAC [5].

```
//-- Localize the object
      std::vector<Point2f> obj;
      std::vector<Point2f> scene;

      for( int i = 0; i < good_matches.size(); i++ )
      {
        //-- Get the keypoints from the good matches
        obj.push_back( keypoints_object[ good_matches[i].queryIdx ].pt );
        scene.push_back( keypoints_scene[ good_matches[i].trainIdx ].pt );
      }

      Mat H = findHomography( obj, scene, CV_RANSAC );

      //-- Get the corners from the image_1 ( the object to be "detected" )
      std::vector<Point2f> obj_corners(4);
      obj_corners[0] = cvPoint(0,0); obj_corners[1] = cvPoint( img_object.cols, 0 );
      obj_corners[2] = cvPoint( img_object.cols, img_object.rows ); obj_corners[3] = cvPoint(
0, img_object.rows );
      std::vector<Point2f> scene_corners(4);

      perspectiveTransform( obj_corners, scene_corners, H);

      //-- Draw lines between the corners (the mapped object in the scene - image_2 )
      line( img_matches, scene_corners[0] + Point2f( img_object.cols, 0), scene_corners[1] +
Point2f( img_object.cols, 0), Scalar(0, 255, 0), 4 );
      line( img_matches, scene_corners[1] + Point2f( img_object.cols, 0), scene_corners[2] +
Point2f( img_object.cols, 0), Scalar( 0, 255, 0), 4 );
      line( img_matches, scene_corners[2] + Point2f( img_object.cols, 0), scene_corners[3] +
Point2f( img_object.cols, 0), Scalar( 0, 255, 0), 4 );
      line( img_matches, scene_corners[3] + Point2f( img_object.cols, 0), scene_corners[0] +
Point2f( img_object.cols, 0), Scalar( 0, 255, 0), 4 );

      //-- Show detected matches
      imshow( "Good Matches & Object detection", img_matches );

    // warp the image. imgB is the image to transform
```

[5] RANSAC (RANdom SAmple Consensus) és un algorisme que troba els "inliers" en un conjunt de dades amb molts "outliers".

```
        Mat outBtrans;
        warpPerspective(img_object, outBtrans, H, cv::Size(img_object.cols, img_object.rows),
cv::INTER_CUBIC);

    imshow ("source", img_object);
    imshow ("source2", img_scene);
    imshow ("output", outBtrans);

    cvWaitKey(0);
```

Càlcul de matriu de transformació per Homografia.

Aquest codi a més presenta les dues imatges en pantalla (principal i secundaria), a més de la imatge final transformada que s'utilitzarà per l'entrenament, traçant unes línies per indicar els punts característics de ambdues imatges i la seva correspondència (figura 29).

A l'esquerra es poden veure les dues imatges originals, amb els punts de coincidència localitzats. A la dreta es pot observar un la imatge transformada a partir de la matriu de transformació obtinguda a l'aplicar la Homografia.

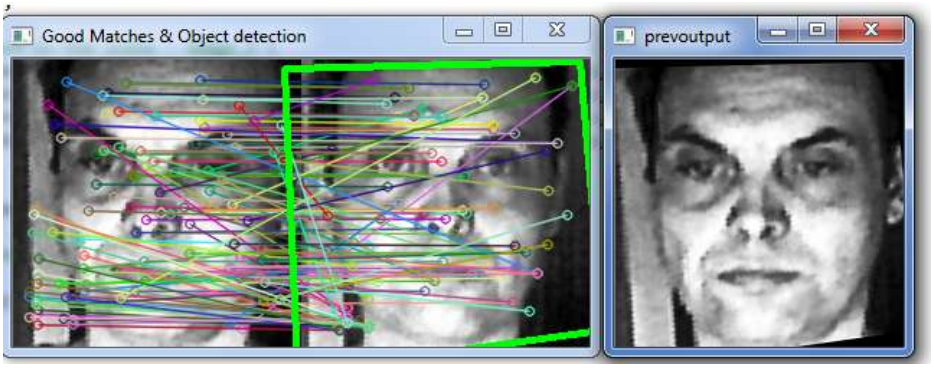


Figura 29: Correspondència de punts de característiques i imatge transformada.

Una vegada complert el vector d'imatges d'entrenament, es crearà un objecte *FaceRecognizer* i s'instanciarà l'entrenament passant com a paràmetres el vector d'imatges i el vector d'etiquetes corresponents, per tal que la funció de reconeixement pugui donar una predicció associada a una etiqueta identificadora d'usuari. L' algoritme de reconeixement utilitzat és un PCA pur, es a dir, té en compte tots els components (eigenfaces) per a la predicció. En el codi següent es pot veure la implementació pràctica d'aquests punts.

```
  // creació objecte FaceRecognizer
  Ptr<FaceRecognizer> model = createEigenFaceRecognizer();

// Entrenament amb vector imatges
  model->train(images, labels);
```

Entrenament.

7.2. Identificació

L'algoritme de reconeixement és similar a l'utilitzat en el punt 6.3.5. explicat anteriorment. A partir de l'objecte *FaceRecognizer* creat i a la complexió de l'entrenament, s'instancia la funció "*predict()*" a la que li passem com a paràmetre

una de les imatges del vector de test. La funció retornarà l'etiqueta del vector labels corresponent a l'usuari predit i un índex de fiabilitat del reconeixement (més fiable quan més alt).

```
int recognize( Mat &test)
{
   // The following line predicts the label of a given test image:

   int predictedLabel = -1;
   double confidence = 0.0;

   model->predict(test, predictedLabel, confidence);

    string result_message = format("Predicted class = %d /confidence = %10.2f.",
predictedLabel, confidence);
    cout << result_message << endl;
    // Here is how to get the eigenvalues of this Eigenfaces model:
    Mat eigenvalues = model->getMat("eigenvalues");
    // And we can do the same to display the Eigenvectors (read Eigenfaces):
    Mat W = model->getMat("eigenvectors");
    // Get the sample mean from the training data
    Mat mean = model->getMat("mean");
    imwrite(format("%s/mean.png", output_folder.c_str()), norm_0_255(mean.reshape(1,
images[0].rows)));

    // Save the Eigenfaces:
    for (int i = 0; i < min(15, W.cols); i++) {
   //for (int i = 0; i < max(10, W.cols-1); i++) {  // working wit as many
Eigenfaces as images
        string msg = format("Eigenvalue #%d = %.5f", i, eigenvalues.at<double>(i));
        //cout << msg << endl;
        // get eigenvector #i
        Mat ev = W.col(i).clone();
        // Reshape to original size & normalize to [0...255] for imshow.
        Mat grayscale = norm_0_255(ev.reshape(1, height));
        // Show the image & apply a Jet colormap for better sensing.
        Mat cgrayscale;
        applyColorMap(grayscale, cgrayscale, COLORMAP_JET);
        //save:
        imwrite(format("%s/eigenface_%d.png", output_folder.c_str(), i),
norm_0_255(cgrayscale));
    }
    // Save the image reconstruction at some predefined steps:
    for(int num_components = min(W.cols, 10); num_components < min(W.cols, 300);
num_components+=15) {
        // slice the eigenvectors from the model
        Mat evs = Mat(W, Range::all(), Range(0, num_components));
        Mat projection = subspaceProject(evs, mean, images[0].reshape(1,1));
        Mat reconstruction = subspaceReconstruct(evs, mean, projection);
        // Normalize the result:
        reconstruction = norm_0_255(reconstruction.reshape(1, images[0].rows));
        // Save:
        imwrite(format("%s/eigenface_reconstruction_%d.png", output_folder.c_str(),
num_components), reconstruction);
    }
    // Display
      imshow("Test Subject", test);

   return 0;
}
```

Reconeixement PCA.

Per finalitzar la funció pot presenta per la consola, el identificador del subjecte predit i l'índex de fiabilitat de la predicció. Amés opcionalment es pot presentar la imatge principal (frontal) del subjecte predit.

Com a complement, es pot també habilitar la gravació dels eigenfaces i la imatge reconstruïda al disc.

7.3. Resultats del test

7.3.1. Condicions del test

Entorn i dades

Per a desenvolupar la fase de test, s'utilitzaran els algorismes descrits en el punt 7 d'aquesta memòria. S'utilitzarà la base de dades especificada al punt 7.1.1. amb un total de 200 imatges d'entrenament i de 50 imatges de test, corresponents a 10 subjectes diferents.

S'efectuaran diferents proves sempre utilitzant un conjunt de 50 imatges de test de 10 subjectes diferents, però variant els conjunts d'entrenament per a cada prova. Serà necessari obtenir resultats per conjunts de 100, 150 y 200 imatges d'entrenament sense utilitzar Homografia. Una vegada finalitzades les tres proves anteriors es repetiran utilitzant en aquest cas Homografia pels conjunts d'entrenament i s'obtindrà una segona matriu de resultats que es compararà amb la de la primera prova.

Mètriques

Les mètriques que s'utilitzaran seran, en primer lloc el percentatge de reconeixement dels subjectes, calculats en base a les prediccions de cada subjecte. Un encert en la predicció comptarà com a 1 i una fallada comptarà com a 0. Per exemple un cas de 5 encerts de 5 comptarà com a un 100% d'encert en la predicció sobre un subjecte determinat.

La segona mètrica que s'utilitzarà serà el índex de fiabilitat de la predicció. Es calcularà la mitjana dels índex de fiabilitat de les prediccions per a cada subjecte tenint en compte que només s'ha de comptar en cas que la predicció hagi estat encertada. En cas contrari, es valorarà com a 0 per tal de calcular la mitjana.

Els resultats finals es calcularan fent la mitjana dels resultats parcials per a cada subjecte. Es confeccionarà una taula amb un resum de resultats parcials i finals per tal de facilitar la interpretació de les dades.

7.3.2. Resultats

A la taula següent (figura 30) es pot observar un resum dels resultats obtinguts a les diferents proves del prototipus de reconeixement facial. A la part esquerra del quadre es presenten els resultats resumits de la identificació facial dels 10 subjectes test, amb diferents conjunts d'entrenament i aplicant o no Homografia per calcular els resultats. El quadre complert de resultats es pot veure a l'annex 11.2 d'aquesta memòria.

	No Homography						Homography					
	Training/test (20/5)		Training/test (15/5)		Training/test (10/5)		Training/test (20/5)		Training/test (15/5)		Training/test (10/5)	
Test user	**Pred. Usr.**	**Confi.**	**Pred. Usr.**	**Confi.**	**Pred. Usr.**	**Confi.**	**Pred. Usr.**	**Confi.**	**Pred. Usr.**	**Confi.**	**Pred. Usr.**	**Confi.**
1	100%	2648	100%	2504	100%	2303	100%	5308	100%	5158	100%	5132
3	100%	4549	100%	5103	100%	5375	100%	6573	80%	5658	100%	6999
4	100%	3790	100%	4245	100%	4751	100%	4728	100%	5382	100%	5338
5	100%	6078	80%	4444	80%	4453	100%	9157	100%	8338	100%	8767
6	100%	5938	100%	6732	100%	6443	100%	7548	100%	6684	100%	7083
8	100%	4330	100%	4180	100%	3905	100%	5181	100%	4481	100%	4911
9	100%	6759	100%	8825	100%	8437	100%	10511	100%	9566	100%	10022
10	100%	7012	100%	9172	100%	8979	100%	9155	100%	8836	100%	9247
11	100%	5145	100%	5789	100%	5697	100%	6265	100%	5678	100%	5870
12	100%	6921	100%	7356	100%	7153	60%	7357	80%	8838	80%	9040
	100%	**5317**	**98%**	**5835**	**98%**	**5750**	**96%**	**7178**	**96%**	**6862**	**98%**	**7241**

Figura 30: taula resum dels resultats de test.

7.3.3. Conclusions del test

Analitzant els resultants podem destacar els següents punts importants :

- Des del punt de vista de l'encert de predicció, els millors resultats obtinguts han estat quan s'ha utilitzat un conjunt d'entrenament de 20 imatges i sense Homografia aplicada, obtenint un índex de predicció del 100% i una fiabilitat de 5317.

- En canvi des del punt de vista de la fiabilitat, els millors resultats obtinguts han estat quan s'ha fet servir un conjunt d'entrenament de 10 imatges i Homografia aplicada, obtenint un índex de reconeixement del 98% i una fiabilitat de 7241.

- L'aplicació d'Homografia als conjunts de test millora sensiblement l'índex de fiabilitat de les prediccions. En els tests efectuats, l'índex de fiabilitat ha millorat una mitjana del 25 % a l'aplicar Homografia.

- En cas de no aplicar Homografia els millors resultats en quant a predicció els obtenim utilitzant un conjunt d'entrenament de 20 imatges, En alguns casos particulars l'aplicació d'Homografia es contraproduent, fent baixar l'índex de reconeixement del 98% al 96%.

- Cal notar que en els resultats obtinguts amb el subjecte nº 12, l'índex de reconeixement s'ha reduït al 73% respecte al 100% obtingut sense aplicar Homografia. Això es degut a la transformació d'algunes imatges del conjunt d'entrenament, ja que en alguns casos la correspondència de punts de característiques no s'efectua correctament i a l'aplicar Homografia podem

obtenir imatges d'entrenament deformades, cosa que dificulta el reconeixement.

Com a conclusions finals podem concloure en primer lloc que utilitzant els algorismes de categorització PCA (amb SVM), amb un entorn de proves controlat, i un conjunt d'entrenament de 200 imatges escollides i prèviament tractades, aconseguim un percentatge de reconeixement molt elevat, entre el 98 i el 100%.

En segon lloc, que aquest percentatge de reconeixement baixa lleugerament quan s'aplica Homografia (un 2%), encara que en general l'índex de fiabilitat del reconeixement puja considerablement fins un 25%.

Per finalitzar, observem que imatges d'entrenament que s'allunyen de la imatge frontal del subjecte, amb el seus ulls mirant a la càmera i expressió neutra, fan que sigui mes feixuga la identificació, així com que sigui contraproduent l'aplicar Homografia, ja que es produeix una deformació de les imatges d'entrenament al trobar-se correlacions errònies als punts de característiques, el que empitjora els resultats del reconeixement.

8. Conclusions i línies de treball futur

La motivació general que ha guiat aquest treball ha estat l'estudi de les diferents tècniques que s'utilitzen avui en per a la identificació facial. En concret, s'ha aprofundit en algunes basades en la combinació d'imatges bidimensionals i dades tridimensionals obtingudes directament mitjançant la captura amb dispositius de vídeo. S'ha completat l'estudi amb la construcció de varis prototipus per tal de validar el funcionament d'algunes d'aquestes tècniques, incidint en l'aportació d'informació tridimensional per tal de fer un reconeixement més acurat.

Per dur a terme el test final s'ha provat d'integrar la captura de punts característics en tres dimensions de la cara d'un subjecte com a base de càlcul de matrius de transformació d'imatges RGB de diferents subjectes per l'entrenament d'un algoritme PCA amb classificador SVM, i així millorar els seus resultats.

S'ha escollit el reconeixement PCA per ser el més comú dels sistemes emprats malgrat que el seu rendiment està molt influenciat per les condicions d'il·luminació, posi, conjunts d'entrenament, etc. Encara i així, els resultants obtinguts han mostrat un percentatge de detecció elevats, però aquest resultats han estat fruit d'un entorn de proves controlat, amb imatges tractades prèviament.

L'aplicació d'un sistema de reconeixement facial en un entorn real com el presentat al principi d'aquesta memòria requerirà uns algorismes hauran de ser més robustos per tal de poder identificar persones amb la precisió i no repudi necessàries per les aplicacions de seguretat que la indústria demanda avui dia, en unes condicions de treball reals i canviants.

En aquesta línia, cal dir que malgrat que el reconeixement facial ha estat un dels objectius principals de la recerca des de la dècade dels 90 del segle passat, l'índex actual de reconeixement en un entorn real està entre el 30 i el 70%, valors clarament insuficients.

En conseqüència, no existeixen en l'actualitat sistemes automàtics de vigilància que es basen en el reconeixement facial, encara que sí que s'estan utilitzant sistemes mixtes de suport, però que a la fi estan validats per essers humans. Per contra i sorprenentment, existeixen línies de recerca que treballen de nou sobre mètodes holístics, concretament sobre una variant de l'algoritme PCA, anomenada LRPCA (Local Region PCA) sobre una evolució de LDA anomenada (CohortLDA) que estan donant resultats prometedors.

Per finalitzar, com a línia futura d'aquest treball, per tal d'obtenir una identificació facial més robusta i amb més precisió, serà necessari aprofundir en mètodes de reconeixement més sofisticats (o una combinació de varis mètodes) capaços de identificat subjectes sota diferents condicions ambientals i diferents punts de vista.

9. Bibliografia i referències

1. **THE BIOMETRICS CONSORTIUM.** (2009). USA Government. *Face recognition* [En línia] http://www.biometrics.gov/Documents/FaceRec.pdf [data consulta: 2/10/2012]

2. **V. ESPINOSA DURÓ** (2004). *Evaluación de sistemas de reconocimiento biométrico*. Escuela Universitaria Politécnica de Mataró. Departamento de Electrónica y automática. [En línea] *http://*www.jcee.upc.es/JCEE2001/PDFs%202000/13ESPINOSA.pdf [data consulta: 2/10/2012]

3. **R. HIETMEYER** (2000) "Biometric identification promises fast and secure processing airline passengers", *The Int'l Civil Aviation Organization Journal*. Vol. 55, n. 9, p. 10-11.

4. **JAIN, A. ROSS, I S. PRABHAKAR**. (2004) "An introduction to biometric recognition". *IEEE Transactions on circuits and systems for video technology 14*, 1, 4–20.

5. **M. TURK I A. PENTLAND** (1991) "Eigenfaces for recognition", *Journal of cognitive neuroscience*. Massachusetts Institute of Technology Vol. 3, n. 1.

6. **FRANCO I L. YAMASAKI.** (2009) *Reconocimiento facial en línea, una arquitectura Open Source para validación de identidad de estudiantes para el LMS MOODLE.* IV Congreso de Cybersociedad 2009. [En línea]. http://www.cibersociedad.net/congres2009/es/coms/. [data consulta: 2/10/2012]

7. **W. ZHAO, R. CHELLAPPA, P.J. PHILLIPS I A.ROSENFELD ,(**2003) "Face recognition: A literature Survey*". ACM Comput*. Surv. 35, 4 (Dec. 2003), p. 399-458.

8. **J. GOLDSTEIN, L. D. HARMON, AND A. B. LESK.** (1971). "Identification of Human Faces," *Proc. IEEE*, May 1971, Vol. 59, No. 5, p. 748-760.

9. **L. SIROVICH I M. KIRBY**. (1987) "A Low-Dimensional Procedure for the Characterization of Human Faces," *J.Optical Soc. Am*. A, 1987, Vol. 4, No.3, p. 519-524.

10. **BOLME, R. BEVERIDGE, M. TEIXEIRA, AND B. DRAPER, (2003).** "The CSU Face Identification Evaluation System: Its Purpose, Features and Structure," *International Conference on Vision Systems, Graz, Austria*, April 1-3, 2003. (Springer-Verlag) p. 304-311.

11. **MIT MEDIA LABORATORY VISION AND MODELING GROUP** (2002), "Photobook/Eigenfaces/Demo" 25 July 2002. [En línea] http://vismod.media.mit.edu/vismod/demos/facerec/basic.html. [data consulta: 11/10/2012]

12. **R.BRUNELLI I T. POGGIO** (1993). "Face recognition: features versus templates", IEEE Transactions on Pattern Analisys and Machine Intelligence, Vol.15, p. 1042-1052.

13. **Y. FREUD I R. SCHAPIRE.** (1999). A Short Introduction to Boosting. http://www.cs.utah.edu/~piyush/teaching/brief_intro_boosting.pdf. [data consulta: 11/10/2012]

14. **P. VIOLA I M. JONES**,(2001) "Rapid object detection using a boosted cascade of simple features," *in Proc. IEEE Conf. Computer Vision & Pattern Recognition, 2001*.

15. **R.A. Fisher** (1936). "The Use of Multiple Measurements in Taxonomic Problems". *Annals of Eugenics* 7 (2): 179–188. http://dx.doi.org/10.1111%2Fj.1469-1809.1936.tb02137.x [data consulta: 17/10/2012]

16. **H. YU, J. YANG.** (2001). "A direct LDA algorithm for high-dimensional data — with application to face recognition", *Pattern Recognition*, 34 (10), 2067–2069.

17. **L. WISKOTT, J.-M. FELLOUS, N. KUIGER, C. VON DER MALSBURG,** (1997). "Face recognition by elastic bunch graph matching". Pattern Analisys and Machine Intelligence. IEEE. Vol. 19 , Issue: 7, p. 775 - 779

18. **M. M. BRONSTEIN, A. M. BRONSTEIN, R. KIMMEL.** (2004). "Three-Dimensional Face Recognition". Department of Computer Science (Technion). Israel Institute of Technology. [En línia]. http://www.cs.technion.ac.il/~ron/PAPERS/BroBroKimIJCV05.pdf [data consulta: 19/10/2012]

19. **L. SMITH.** (2002). "A tutorial on Principal Component Analisys." [En línia]. http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf. [data consulta: 20/10/2012]

20. **R. LIENHART, J. MAYDT.** (2002) An Extended Set of Haar-like Features for Rapid Object Detection. IEEE ICIP 2002, Vol. 1, pp. 900-903, Sep. 2002.

21. **P. WAGNER** (2003) An Extended Set of Haar-like Features for Rapid Object Detection. [En línia]. http://www.bytefish.de/blog/face_recognition_with_opencv2. [data consulta: 9/11/2012]

10. Altres treballs consultats

A. **E. CABELLO.** (2004). *Técnicas de reconocimiento facial mediante redes neuronales. Tesis Doctoral*. Departamento de Tecnología Fotónica. Facultad de Informática (UPM)[En línea]. http://oa.upm.es/215/1/10200404.pdf . [Data consulta: 10/10/2012]

B. **C. REYES** (2005). *Reconocimiento facial mediante visión artificial. Proyecto de fin de carrera*. Escuela técnica superior de Ingenieros. Universidad de Sevilla. [En línea].

http://bibing.us.es/proyectos/abreproy/11025/fichero/Reconocimiento+Facial+...Artificial.pdf. [Data consulta: 25/09/2012]

A. **Huang J.** (2002). Component-based Face Recognition with 3D Morphable Models. Department of Electrical Engineering and Computer Science (MIT).)[En línea]. http://cbcl.mit.edu/cbcl/publications/theses/thesis-huang.pdf. [Data consulta: 7/10/2012]

11. Annexes

11.1. Pla de treball

Reconeixement facial utilitzant llibreries d'ús públic

Resum
El propòsit del projecte és el coneixement de l'estat de l'art i els darrers avenços de les tècniques de reconeixement facial mitjançant visió artificial, així com l'anàlisi i comparació de les diverses tècniques utilitzades i, finalment, el desenvolupament d'un prototipus utilitzant llibreries d'ús públic (OpenCV 2).

Introducció
La disciplina de la visió per computador (o visió artificial), tracta del desenvolupament d'algorismes que reprodueixen una de les capacitats d'inferència més importants del cervell humà: la d'extreure propietats del mon extern mitjançant la imatge dels objectes que capta l'ull.

Aquestes propietats permeten una comprensió de la imatge en termes d'extracció de la seva informació simbòlica, utilitzant models construïts amb l'ajuda de la geometria, la física, l' estadística, i la teoria de l'aprenentatge.

Com a disciplina científica, la visió artificial compren la teoria que hi ha darrere dels sistemes artificials que extreuen informació de les imatges. Les dades d'una imatge poden prendre moltes formes, com ara seqüències de vídeo, vistes de càmeres múltiples, o multidimensionals.
Com a disciplina tecnològica, la Visió per Artificial cerca l'aplicació de les teories i models a la construcció de sistemes de visió per computador. Alguns exemples d'aquestes aplicacions poden ser sistemes per:

- Control de processos.
- Navegació.
- Detecció d'esdeveniments (p.e. vigilància).
- Modelat d'objectes o entorns.
- Interacció amb essers humans.

- Inspecció automàtics (p.e. en cadenes de producció).
- Identificació.
- Etc.

Objectius

L'objectiu principal d'aquest treball és l'estudi dels diferents algoritmes i tècniques emprades pel reconeixement de cares dins de la disciplina de la Visió per Computador.

Un altre objectiu, el secundari, és el dissenyar i desenvolupar un prototipus de programari que implementi alguna de les tècniques actuals del reconeixement facial, utilitzant llibreries de codi obert.

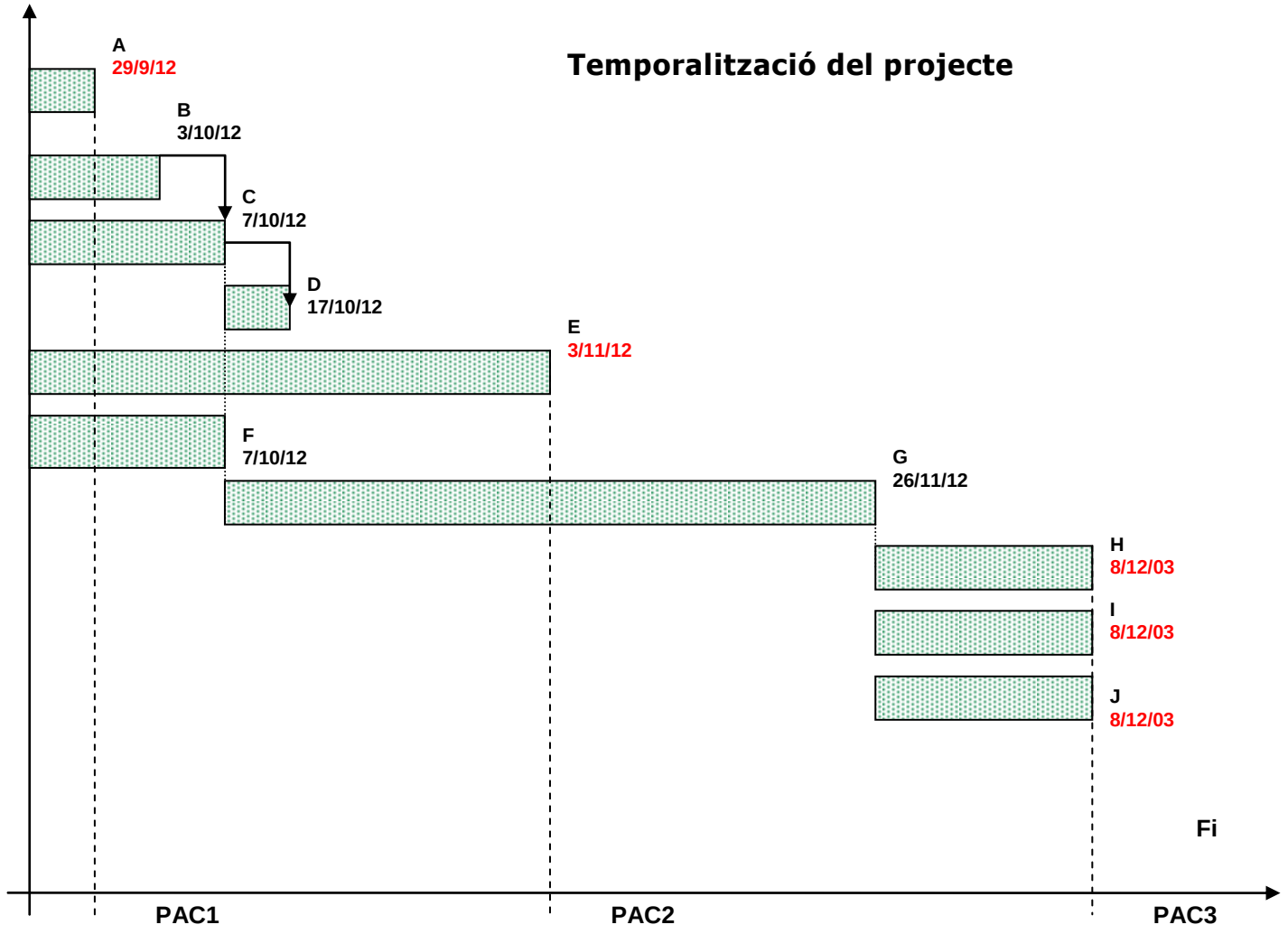
Índex preliminar de la memòria

1. Introducció i planificació
2. Objectius
3. Estat de l'Art de la Visió per computador
   3.1. Biometria
   3.2. Generalitats de la Visió per computador
   3.3. Conceptes Bàsics
   3.4. Antecedents
4. Tècniques de reconeixement facial
   4.1. Algoritmes
   4.2. Mètodes
5. Llibreries d'ús públic: OpenCV2
   5.1. Entorn de Programació
   5.2. Funcions implementades
6. Disseny i implementació del prototipus
   6.1. Menú principal
   6.2. Detecció de cares
   6.3. Segmentació Imatges
   6.4. Extracció de característiques facials
   6.5. Identificació de múltiples subjectes. Classificació
   6.6. Proves i resultats
7. Conclusions i línies de treball a futur
8. Bibliografia i referències
9. Annexes

Detall de les tasques

Id.	Tasca	Durada	Inici	Lliurament	Precedents
A	Introducció i planificació – Pla de treball	5	19/9/2012	29/9/2012	-
B	Objectius i organització de la memòria	5	26/9/2012	3/10/2012	-
C	Estat de l'art de la Visió per Computador	8	26/9/2012	7/10/2012	B
D	Tècniques de reconeixement facial	10	7/10/2012	17/10/2012	B
E	Llibreries d'ús públic	15	26/9/2012	3/11/2012	B
F	Entorn de programació	30	26/9/2012	7/10/2012	B
G	Disseny i Implementació del prototipus	50	7/10/2012	26/11/2012	E,F
H	Conclusions i línies de treball a futur	12	26/11/2012	8/12/2012	G
I	Bibliografia i referències	12	26/11/2012	8/12/2012	H
J	Annexes	12	26/11/2012	8/12/2012	I

Nota: dates en **color blau** corresponen a lliuraments de l'avaluació continuada.


Temporalització del projecte
A
29/9/12
B
3/10/12
C
7/10/12
D
17/10/12
E
3/11/12
F
7/10/12
G
26/11/12
H
8/12/03
I
8/12/03
J
8/12/03
Fi
PAC1
PAC2
PAC3

11.2. Resultats del test

	No Homography						Homography					
	Training/test (20/5)		Training/test (15/5)		Training/test (10/5)		Training/test (20/5)		Training/test (15/5)		Training/test (10/5)	
Test user	Predicted Usr.	Confid.	Predicted Usr.	Confid.	Predicted Usr.	Confid.	Predicted Usr.	Confid.	Predicted Usr.	Confid.	Predicted Usr.	Confid.
1	1	3510	1	3417	1	3106	1	3515	1	2824	1	3397
1	1	6172	1	5860	1	5465	1	6019	1	5412	1	5925
1	1	3558	1	3243	1	2943	1	3901	1	5923	1	3851
1	1	0	1	0	1	0	1	6482	1	5680	1	6185
1	1	0	1	0	1	0	1	6622	1	5952	1	6301
1	**100%**	**2648**	**100%**	**2504**	**100%**	**2303**	**100%**	**5308**	**100%**	**5158**	**100%**	**5132**
3	3	4049	3	5008	3	5112	3	5236	3	7378	3	5721
3	3	3359	3	4619	3	5194	3	5474	3	7292	3	6346
3	3	4035	3	4424	3	5521	3	4940	3	7055	3	6162
3	3	6069	3	5970	3	5364	3	10460	0	0	3	10076
3	3	5231	3	5494	3	5685	3	6754	3	6563	3	6689
3	**100%**	**4549**	**100%**	**5103**	**100%**	**5375**	**100%**	**6573**	**80%**	**5658**	**100%**	**6999**
4	4	4954	4	5542	4	4686	4	5379	4	5031	4	5498
4	4	3868	4	4625	4	3992	4	4371	4	3379	4	3902
4	4	0	4	0	4	4191	4	4386	4	6597	4	5147
4	4	5448	4	5394	4	5059	4	4985	4	6449	4	6222
4	4	4679	4	5662	4	5829	4	4520	4	5456	4	5919
4	**100%**	**3790**	**100%**	**4245**	**100%**	**4751**	**100%**	**4728**	**100%**	**5382**	**100%**	**5338**
5	5	7926	5	7411	5	7117	5	10798	5	9929	5	10304
5	5	7071	0	0	0	0	5	13119	5	12128	5	12609

5	5	4797	5	4640	5	5241	5	6794	5	6122	5	6484
5	5	5329	5	5099	5	5204	5	6404	5	5675	5	6081
5	5	5269	5	5069	5	4702	5	8672	5	7838	5	8357
5	**100%**	**6078**	**80%**	**4444**	**80%**	**4453**	**100%**	**9157**	**100%**	**8338**	**100%**	**8767**
6	6	8502	6	7970	6	7597	6	11013	6	10126	6	10679
6	6	6856	6	7875	6	7478	6	8493	6	7771	6	8040
6	6	3797	6	6787	6	6463	6	6634	6	5563	6	5997
6	6	4238	6	4015	6	3858	6	4805	6	4162	6	4455
6	6	6298	6	7011	6	6821	6	6796	6	5799	6	6244
6	**100%**	**5938**	**100%**	**6732**	**100%**	**6443**	**100%**	**7548**	**100%**	**6684**	**100%**	**7083**
8	8	5243	8	5017	8	4627	8	6409	8	5186	8	6018
8	8	6770	8	6849	8	6210	8	8540	8	7243	8	7664
8	8	0	8	0	8	0	8	0	8	0	8	0
8	8	5060	8	4578	8	4432	8	4688	8	4324	8	4570
8	8	4576	8	4457	8	4256	8	6267	8	5651	8	6302
8	**100%**	**4330**	**100%**	**4180**	**100%**	**3905**	**100%**	**5181**	**100%**	**4481**	**100%**	**4911**
9	9	7538	9	8516	9	8141	9	10220	9	9396	9	9802
9	9	6585	9	8468	9	8088	9	10468	9	9302	9	9792
9	9	7299	9	8384	9	8014	9	10249	9	9459	9	9816
9	9	5556	9	9762	9	9318	9	11234	9	10263	9	10778
9	9	6818	9	8996	9	8624	9	10385	9	9409	9	9922
9	**100%**	**6759**	**100%**	**8825**	**100%**	**8437**	**100%**	**10511**	**100%**	**9566**	**100%**	**10022**
10	10	10958	10	10678	10	9912	10	11207	10	10108	10	10527
10	10	8502	10	8351	10	7963	10	8498	10	7755	10	8307
10	10	0	10	10368	10	9987	10	9494	10	9435	10	10147
10	10	6839	10	8140	10	7850	10	6971	10	7723	10	8103
10	10	8761	10	8325	10	9182	10	9603	10	9158	10	9150
10	**100%**	**7012**	**100%**	**9172**	**100%**	**8979**	**100%**	**9155**	**100%**	**8836**	**100%**	**9247**

11	11	4841	11	6467	11	6107	11	7433	11	6173	11	6636
11	11	5588	11	6442	11	6688	11	6362	11	5995	11	6452
11	11	5027	11	4994	11	4452	11	4914	11	4245	11	4601
11	11	4754	11	6059	11	5860	11	6619	11	5832	11	5430
11	11	5516	11	4981	11	5380	11	5996	11	6145	11	6231
11	**100%**	**5145**	**100%**	**5789**	**100%**	**5697**	**100%**	**6265**	**100%**	**5678**	**100%**	**5870**
12	12	6541	12	7829	12	8493	0	0	12	11406	12	10880
12	12	6499	12	7192	12	6318	12	12983	12	11791	12	12397
12	12	9856	12	9936	12	10366	0	0	0	0	0	0
12	12	4259	12	4090	12	3799	12	10675	12	9070	12	10135
12	12	7449	12	7734	12	6791	12	13126	12	11922	12	11790
12	**100%**	**6921**	**100%**	**7356**	**100%**	**7153**	**60%**	**7357**	**80%**	**8838**	**80%**	**9040**

11.3. Algoritme emprat al prototipus

S'adjunta fitxer "JoanHermanPFC.cpp" amb el codi principal del projecte.