DETECTARE ŞI URMĂRIRE AUTOMATĂ A FEŢELOR UTILIZÂND IMPLEMENTĂRI OPENCV ALE METODEI VIOLA-JONES

Mihnea Horia Vrejoiu mihnea@dossv1.ici.ro

Ştefan Alexandru Preda stefanalex@ici.ro

Mădălina Zamfir madalina@ici.ro

Institutul Naţional de Cercetare-Dezvoltare în Informatică, ICI - Bucureşti

Rezumat: S-a proiectat, implementat şi testat un sistem experimental de detecţie şi urmărire a feţelor umane în imagini statice, respectiv dinamice – cadre video succesive din fişiere video (filme), sau capturate în timp real (live) cu o cameră video sau chiar cu una web – bazat pe biblioteca open source de funcţii de vedere artificială (Computer Vision) a Intel, OpenCV. În acest context, s-a imaginat, proiectat şi implementat un scenariu, o metodă şi un algoritm de detecţie şi urmărire (tracking) a fiecărei feţe statice sau în mişcare identificate ca atare în cadrele respective, prin etichetarea / indexarea acesteia şi cu calcularea poziţiilor viitoare posibile. Au fost elaborate soluţii pentru etichetarea / indexarea corectă a feţelor în cadrele succesive inclusiv în situaţiile de posibile ambiguităţi în identificarea corectă a acestora în cazul „pierderii" lor temporare din cadre şi reapariţiei ulterioare.

Cuvinte cheie: detecţia feţelor, urmărirea feţelor, vedere artificială, analiză de imagini, cascadă Haar, OpenCV.

Abstract: An experimental system has been designed, implemented and tested for human face detection and tracking in still, respectively dinamic images – succesive video frames from video files (movies) or captured in real time (live) with a video or even a web camera – based on the Intel's open source computer vision functions library, OpenCV. In this context, there have been imagined, designed and implemented a scenario, a method and an algorithm for detecting and tracking each static or moving face thus identified in the respective frames, by tagging (indexing) it and computing its future possible positions. There have been provided solutions for correctly tagging / indexing the faces in successive frames including those possible situations of ambiguity in their correct identification in the case of temporarily „loosing" them from some frames and further reappearence.

Keywords: Face Detection, Face Tracking, Computer Vision, Image Analysis, Haar Cascade, OpenCV.

1. Introducere

În contextul unui proiect [1] derulat în cadrul Programului Nucleu la ICI, s-a proiectat şi implementat între altele şi un sistem experimental destinat detecţiei şi/sau urmăririi figurilor umane în imagini statice sau respectiv dinamice – cadre video succesive din fişiere video (filme), sau capturate în timp real (live) cu o cameră video sau chiar cu una web – bazat pe biblioteca open source de funcţii de vedere artificială (Computer Vision) a Intel, OpenCV.

Sistemul experimental proiectat a avut drept scop iniţial testarea funcţiilor care implementează în biblioteca OpenCV [2] metoda Viola-Jones [3, 4, 5] de detecţie a feţelor umane, extinsă pentru poziţii / orientări diferite ale acestora [6, 7], atât în imagini statice (încărcate din fişiere imagine sau capturate prin „îngheţare" din flux video continuu cât şi în timp real din cadre video live obţinute prin achiziţie video continuă sau dintr-un film „înregistrat" anterior.

În cazul celor din urmă s-a imaginat, proiectat şi implementat şi un scenariu, o metodă şi un algoritm de urmărire (tracking) a fiecărei feţe statice sau în mişcare, identificate ca atare în cadrele respective, prin etichetarea / indexarea acesteia. Au fost elaborate soluţii pentru etichetarea / indexarea corectă a feţelor în cadrele succesive, inclusiv pentru situaţiile de posibile ambiguităţi în etichetarea corectă a acestora în cazul „pierderii" lor temporare din cadre.

Pentru fiecare faţă detectată într-un cadru sunt calculate poziţiile viitoare posibile. Acestea sunt utilizate atât pentru identificarea corectă şi coerentă în cadrul următor cât şi pentru cazurile în care faţa respectivă „dispare" dintr-un motiv sau altul din cadrele video pentru o perioadă scurtă de timp – nu este detectată, se întoarce cu spatele la cameră, iese temporar din scenă sau este obturată temporar de alt obiect sau altă persoană etc. – şi apoi reapare.

În proiectarea şi implementarea scenariului şi algoritmului de urmărire a feţelor în flux video continuu, s-a pornit de la o euristică de tipul: între două cadre video succesive, o aceeaşi faţă nu poate apărea decât cel mult într-o anumită vecinătate nu foarte extinsă faţă de localizarea iniţială.

Astfel, în mod normal, o față va fi etichetată / indexată similar, cu aceeaşi etichetă (acelaşi index) în două cadre succesive, dacă centrele geometrice ale dreptunghiurilor de încadrare în cele două cadre nu sunt mai depărtate decât o anumită distanță maximă stabilită empiric.

Pentru fiecare față detectată într-un cadru se poate defini o anumită regiune în care ne putem aştepta ca ea să fie regăsită în cadrul următor, respectiv centrul geometric al dreptunghiului său de încadrare în acest nou cadru se va regăsi cel mai probabil într-un cerc de rază egală cu distanța maximă amintită mai înainte şi centrat în poziția corespunzătoare centrului său geometric din cadrul anterior. În cazul mai multor fețe, corespondentul fiecăreia în cadrul nou va fi dat de distanța minimă între centrele geometrice respective. Eventuale situații de ambiguitate în cazul mai multor fețe apropiate sunt rezolvate prin găsirea distanței minime pentru fiecare dintre acestea la posibilii corespondenți din celălalt cadru. Totodată, pentru eliminarea ambiguităților ce pot apărea în cazul intersectării mai multor persoane, când unele din fețe sunt obturate temporar de altele, s-a utilizat euristica dată de dimensiunile dreptunghiurilor de încadrare ale fețelor care datorită profunzimii sunt mai mari pentru cele situate în planuri mai apropiate şi mai mici pentru cele mai îndepărtate. Pe baza acestui criteriu se poate păstra etichetarea / indexarea corectă a fiecărei fețe după reapariția în cadru a celor temporar obturate.

2. Descrierea metodei şi algoritmilor pentru detecția şi urmărirea fețelor

În linii mari, algoritmul utilizat pentru detectarea fețelor într-o imagine (într-un cadru video) este următorul:

1. Crearea vectorului ce va conține vectorii cu informații despre fiecare față detectată: patru coordonate de încadrare plus alte variabile de tip întreg care cuantifică numeric răspunsurile la următoarele întrebări:
    a. A fost afişat dreptunghiul de încadrare pentru fața respectivă?
    b. Care este indexul feței respective?
    c. Care este sensul de mişcare a feței respective?
    d. Care este distanța parcursă în imaginea video de fața respectivă?
2. Încărcarea instrumentelor OpenCV, cum sunt cascadele pentru detectarea fețelor; pornirea camerei video / web, a unui fişier video sau încărcarea unei imagini din fişier.
3. Analiza imaginii sau a fiecărui cadru (*frame*) video. În această etapă sunt parcurşi următorii paşi:
    a. Preluarea conținutului efectiv al imaginii într-o matrice;
    b. Transformarea acesteia în 256 de nivele de gri (format *greyscale*);
    c. Detectarea fețelor;
    d. Numărarea fețelor detectate şi preluarea informațiilor asociate lor în vectorul creat la pasul 1.
4. Afişarea rezultatului pe ecranul monitorului, prin desenarea dreptunghiurilor (pătratelor) de încadrare în jurul fețelor detectate.

În proiectarea şi implementarea scenariului şi algoritmului de urmărire a fețelor în flux video continuu s-a pornit de la o euristică de tipul: între două cadre video succesive o aceeaşi față nu poate apărea decât cel mult într-o anumită vecinătate nu foarte extinsă față de localizarea inițială.

Pe baza acestui criteriu, în mod normal, o față va fi etichetată / indexată similar, cu aceeaşi etichetă (acelaşi index) în două cadre succesive dacă centrele geometrice ale dreptunghiurilor de încadrare în cele două cadre nu sunt mai depărtate decât o anumită distanță maximă stabilită empiric. Astfel, pentru fiecare față detectată într-un cadru se poate defini o anumită regiune în care ne putem aştepta ca ea să fie regăsită în cadrul următor, respectiv centrul geometric al dreptunghiului său de încadrare în acest nou cadru se va regăsi cel mai probabil într-un cerc de rază egală cu distanța maximă amintită mai înainte şi centrat în poziția corespunzătoare centrului său

geometric din cadrul anterior. În cazul mai multor fețe, corespondentul fiecăreia în cadrul nou va fi dat de distanța minimă între centrele geometrice respective. Eventuale situații de ambiguitate în cazul mai multor fețe apropiate sunt rezolvate prin găsirea distanței minime pentru fiecare dintre acestea la posibilii corespondenți din celălalt cadru.

Au mai fost avute în vedere următoarele probleme care ar putea apărea:

- Cum se pot urmări (eticheta / indexa) corect mai departe persoanele care se intersectează cu alte persoane, respectiv fețele care sunt temporar obturate / mascate de alte fețe? Pe baza observării modului în care detectorul de fețe din biblioteca OpenCV utilizat creează dreptunghiurile de încadrare a fețelor detectate (mai mari pentru fețele din planurile mai apropiate), această problemă a fost împărțită în două situații:

    a) când persoana mai apropiată de cameră trece prin dreptul unei persoane mai îndepărtate de cameră. Pentru a se realiza în continuare urmărirea facială corectă în acest caz, în program se organizează (ordonează) vectorul de vectori cu informații asociate fețelor în funcție de mărimea dreptunghiului fiecărei fețe descoperite.

    b) când o persoană mai îndepărtată de cameră trece prin spatele unei alte persoane sau obiect aflate mai aproape de cameră. În acest caz se calculează posibile poziții viitoare ale feței ce trece prin spatele alteia, astfel încât, când aceasta va reapărea în video, poziția sa va fi comparată cu pozițiile prevăzute, putându-se stabili dacă este vorba despre o persoană nou apărută, sau este tot cea urmărită anterior.

- Din punctul de vedere al implementării, o posibilă problemă ar reieşi chiar din cele descrise mai sus. Astfel, din cauza faptului că se creează un număr de vectori cu informații despre posibilele poziții viitoare ale fețelor ce trec prin spatele unui obiect/persoană sau ies temporar din imagine este posibilă încărcarea exagerată a memoriei cu informații istorice acumulate inutile. Din acest motiv, vectorii ce oferă informații cu previziuni asupra pozițiilor viitoare au fost indexați corespunzător şi, la fiecare ciclu de recunoaştere a fețelor se şterg din memorie, iar apoi se creează din nou.

În cazul urmăririi fețelor într-o secvență de cadre video, pasul 3 din algoritmul de detecție a fețelor prezentat mai sus se completează cu parcurgerea următorilor paşi şi se repetă ciclic atâta timp cât există cadre noi (flux video):

1. Compararea numărului de fețe descoperite în cadrul actual cu cel din cadrul precedent.
    a. În cazul în care numărul de fețe este mai mic decât în cadrul anterior se caută care dintre fețele găsite anterior nu sunt prezente în cadrul actual şi li se atribuie informația respectivă.
    b. În cazul în care numărul de fețe este mai mare decât în cadrul anterior se compară numărul de fețe găsite cu numărul maxim de fețe găsite până în momentul respectiv şi, după caz, ori se creşte numărul maxim de fețe găsite, ori se caută posibile fețe găsite anterior ce pot avea pozițiile viitoare calculate pe locul celor noi găsite şi se rescrie informația asociată acestora.
2. Organizarea vectorului de vectori cu informații asociate fețelor în funcție de poziția acestora în imagine şi în funcție de informația din care reiese dacă încadrarea acestora a fost sau nu afişată.
3. Calcularea unor posibile poziții viitoare ale fețelor detectate.
4. Calcularea şi afişarea sensului de deplasare şi a distanței parcurse de către fețele respective.

De asemenea, la pasul 4 al algoritmului de detecție nu se mai reprezintă în acest caz numai dreptunghiurile de încadrare ci şi etichetele (indecşii) de identificare pentru fiecare față detectată.

Fiecare index de etichetare a fețelor detectate este atribuit secvențial cronologic, în funcție de momentul (cadrul) în care s-a detectat fața respectivă. În situația în care într-un acelaşi cadru apar mai multe fețe noi, care nu au mai fost detectate anterior, ordinea indexării acestora este de la stânga la dreapta.

3. Mediul şi instrumentele de implementare şi testare

Pentru implementarea sistemului experimental s-a folosit mediul de dezvoltare integrat NetBeans împreună cu compilatorul MinGW pentru limbajul de programare C++ şi cu biblioteca OpenCV.

NetBeans este un mediu de dezvoltare integrat (IDE) pentru dezvoltarea programelor în oricare din limbajele de programare Java, PHP, C/C++, precum şi HTML5. NetBeans IDE este creat în limbajul de programare Java şi poate rula pe Windows, OS X, Linux, Solaris şi alte platforme care suportă o maşină virtuală Java (*Java Virtual Machine* – JVM) compatibilă. Platforma NetBeans permite ca aplicaţiile să fie dezvoltate dintr-un set de componente software modulare (module). Aplicaţii bazate pe platforma NetBeans (inclusiv *NetBeans IDE*) pot fi modificate, după caz, de către terţi dezvoltatori.

MinGW (*Minimalist GNU* pentru Windows), numit anterior mingw32, este o distribuţie a colecţiei de compilatoare GNU (GCC) si GNU binutils pentru a fi utilizate în elaborarea de aplicaţii native Microsoft Windows. Ea conţine un set de fişiere antet (*header*) distribuite şi distribuibile gratuit şi biblioteci de import statice, pentru a permite utilizarea interfeţei de programare Windows (Windows API).

OpenCV (*Open Source Computer Vision Library*) este o bibliotecă, dezvoltată iniţial de Intel, care conţine funcţii de programare dedicate vederii artificiale, de captură / manipulare, prelucrare / procesare şi analiză de imagini în timp real. Aceasta poate fi utilizată gratuit, sub licenţă de tip *open source*. Biblioteca este de tip multiplatformă (*cross-platform*).

Pentru a facilita funcţionarea celor trei componente împreună, biblioteca OpenCV a fost întâi compilată cu programul **CMake**. Acesta este un program software *free*, *cross-platform*, pentru gestionarea procesului de construcţie a software-ului folosind o metodă de compilator independent. Acesta are, de asemenea, dependenţe minime, necesitând doar un compilator de C++ pe propriul său sistem de *build*.

Am precizat deja faptul că partea de manipulare şi prelucrare / procesare / analiză de imagini se face prin funcţionalităţile oferite de biblioteca OpenCV. Aceasta se realizează concret prin apelarea unor funcţii din anumite DLL-uri ale OpenCV, respectiv:

```
libopencv_core.dll,
libopencv_highui.dll,
libopncv_imgproc.dll
libopencv_objdetect.dll.
```

De asemenea, pentru detecţia feţelor sunt folosite următoarele două cascade Haar din OpenCV:

```
haar_cascade_profileface.xml
haar_cascade_frontalface_alt.xml.
```

Prima cascadă permite detectarea feţelor din profil, iar cea de-a doua detectarea feţelor în vedere frontală.

Trebuie să facem aici observaţia că, deşi mai există un număr de cascade implementate în OpenCV pentru detectarea feţelor din profil sau frontale, din experimentele şi testele efectuate cu sistemul experimental s-a constatat că cele două menţionate mai sus sunt cele mai eficiente. De exemplu, la testarea unei alte cascade pentru detecţia frontală, numită

```
haar_cascade_frontalface_default.xml
```

s-a observat că, pe aceleaşi imagini şi în aceleaşi condiţii, aceasta din urmă are un număr de erori mai mare. Cu celelalte două menţionate mai înainte, performanţele în ce priveşte rata de succes şi viteza de calcul sunt remarcabile, cu rare scăpări, explicabile totuşi.

Din DLL-urile OpenCV utilizate şi menţionate mai sus, sistemul experimental apelează următoarele funcţii / comenzi ce merită să fie amintite aici:

- `CascadeClassifier` - aceasta este o clasă prin care se creează un obiect pentru încărcarea unei cascade;
- `CvNamedWindow` - această comandă creează o fereastră în care vor fi afişate rezultatele prelucrării imaginilor;
- `CvCapture` şi `CvCaptureProperty` - sunt comenzi folosite pentru capturarea imaginilor din fişiere video sau încărcarea lor din fişiere imagine;
- `detectMultiScale` - este comanda pentru detectarea feţelor dintr-o imagine / cadru şi salvarea într-un vector a coordonatelor dreptunghiurilor (pătratelor) de încadrare a acestora, ce vor fi apoi desenate pe imagine în jurul feţelor detectate.

4. Rezultate experimentale

Experimentele s-au desfăşurat în condiţii ambientale diferite în ceea ce priveşte iluminarea şi tonalitatea încăperii, iar achiziţia imaginilor în flux continuu s-a realizat cu ajutorul unei camere web simple.

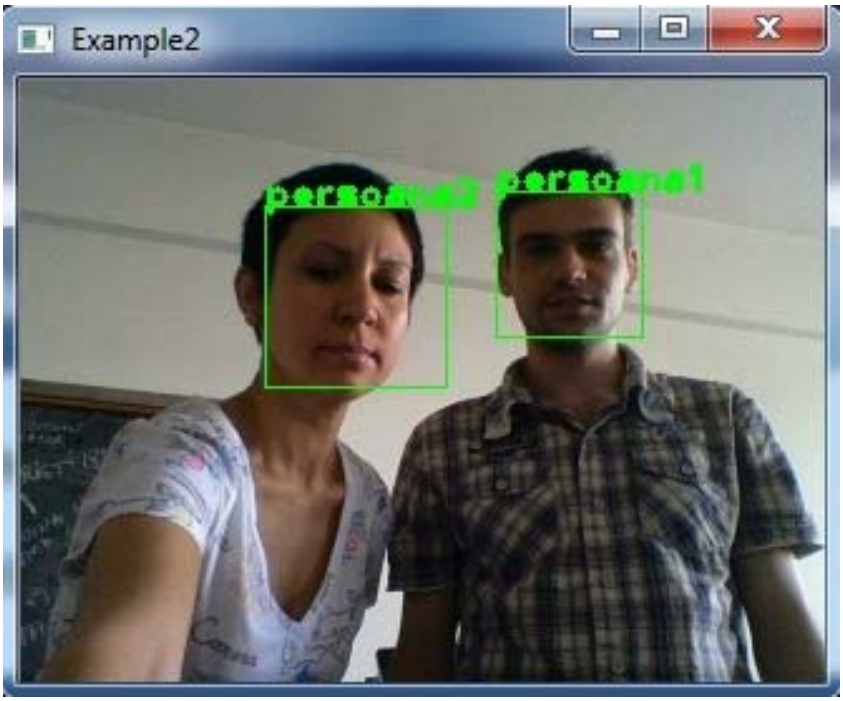
S-a constatat că sporadic mai apar anumite scăpări ale funcţiilor OpenCV utilizate la detectarea unor feţe în unele cadre, mai cu seamă în medii cu luminozitate (prea) mare. Totuşi, algoritmul de etichetare reuşeşte să depăşească cu succes de cele mai multe ori astfel de situaţii. Utilizându-se implementări de casacade Haar diferite din OpenCV se obţin rezultate diferite.

Sistemul experimental implementat poate analiza în timp real cadre cu o rezoluţie de 320×240 pixeli din flux video continuu la 15 cadre pe secundă. Timpul mediu necesar rulării unui ciclu de urmărire facială pe un cadru este în jur de 170 ms pe un calculator (laptop) de generaţie actuală, cu procesor i3 la 2,28 GHz şi 4GB RAM.

Inserăm mai jos câteva imagini sugestive pentru modul de funcţionare al algoritmului de urmărire a feţelor (*face tracking*) propus şi implementat.

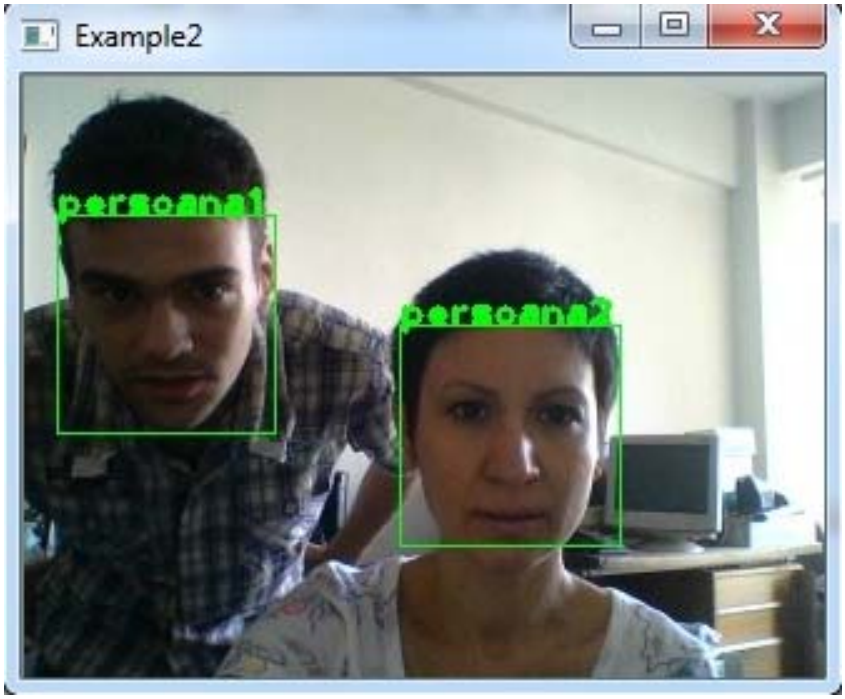
Se poate observa etichetarea / indexarea corectă, în ordinea apariţiei, a feţelor detectate în cadrele respective, precum şi păstrarea etichetării / indexării corecte inclusiv ulterior unei situaţii de dispariţie temporară, prin obturare de către cealaltă, a uneia dintre acestea.

Imaginea 1. – O faţă detectată şi etichetată



Imaginea 2. – Două fețe detectate şi etichetate

Imaginea 3. – Una dintre fețe obturată de cealaltă care este etichetată corect



Imaginea 4. – Fața anterior obturată reapare corect etichetată

5. Concluzii

A fost imaginat, proiectat, implementat şi testat cu succes un sistem experimental pentru detecția şi urmărirea fețelor în imagini utilizându-se implementări ale metodei Viola-Jones din biblioteca *open source* de funcții de vedere artificială a Intel, OpenCV.

Dezvoltarea programelor s-a realizat în limbajul C++.

Sistemul lucrează în timp real, făcând față fără probleme la achiziție video continuă cu o cameră web simplă, la o rezoluție de 320×240 pixeli, cu 15 cadre pe secundă.

Algoritmul de etichetare (indexare) a fețelor detectate în cadrele video lucrează corect, păstrând identificarea acestora inclusiv în situațiile de posibile ambiguități datorate „dispariției" temporare a unora dintre ele şi reapariției lor ulterioare.

BIBLIOGRAFIE

1. **VREJOIU, M. H.; HOTĂRAN, A. M.; PREDA, Ş.; ZAMFIR, M.:** Metode, tehnici şi algoritmi pentru localizarea şi urmărirea figurilor umane în imagini statice, respectiv dinamice. Studiu. Specificații de definire metode, tehnici şi algoritmi. Specificații de implementare metode, tehnici şi algoritmi. Sistem experimental – Raport de cercetare, etapa 1 (unică) a proiectului PN09230606, ICI Bucureşti, iulie 2013.
2. **HEWITT, R.:** Seeing With OpenCV, Part 2: Finding Faces in Images, in SERVO Magazine, T & L Publications Inc., February 2007.
3. **VREJOIU, M. H.; HOTĂRAN, A. M.:** Detectarea automată a fețelor umane. Metoda Viola-Jones, în Revista Română de Informatică şi Automatică (RRIA), vol.23, nr.2, iunie 2013.
4. **VIOLA, P.; JONES, M.:** Rapid Object Detection using a Boosted Cascade of Simple Features, in the Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), 2001.

5. **VIOLA, P.; JONES, M. J.:** Robust Real-Time Face Detection, in the International Journal of Computer Vision (IJCV) 57(2), Kluwer Academic Publishers, 2004, pp. 137-154.
6. **JONES, M.; VIOLA, P.:** Fast multi-view face detection, in Technical report, Mitsubishi Electric Research Laboratories, TR2003-96, 2003.
7. **LIENHART, R.; MAYDT, J.:** An extended set of Haar-like features for rapid object detection, in Proc. of ICIP, 2002.