Technical University of Cluj - Napoca
Computer Science Department

Interfete om-calculator

Curs 5-6

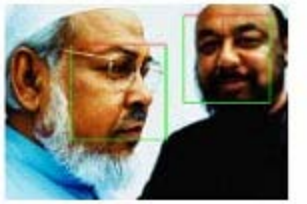
Detectia feţelor.

Detectia componentelor faciale (statica si dinamica)

Scop

Identificarea fetelor umane si localizarea lor in imagine, indiferent de:

- Pozitie
- Scala
- Rotatie (in planul imaginii)
- Orientare (rotatie in afara planului imaginii)
- Iluminare



gure 3: Samples of detection results of faces of various poses

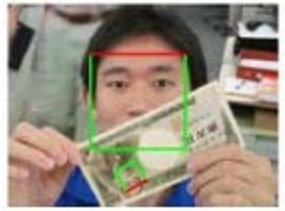


Figure 1: Detection results of occluded fa

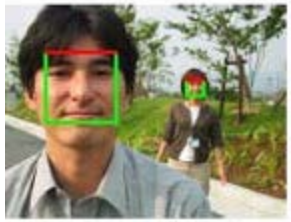
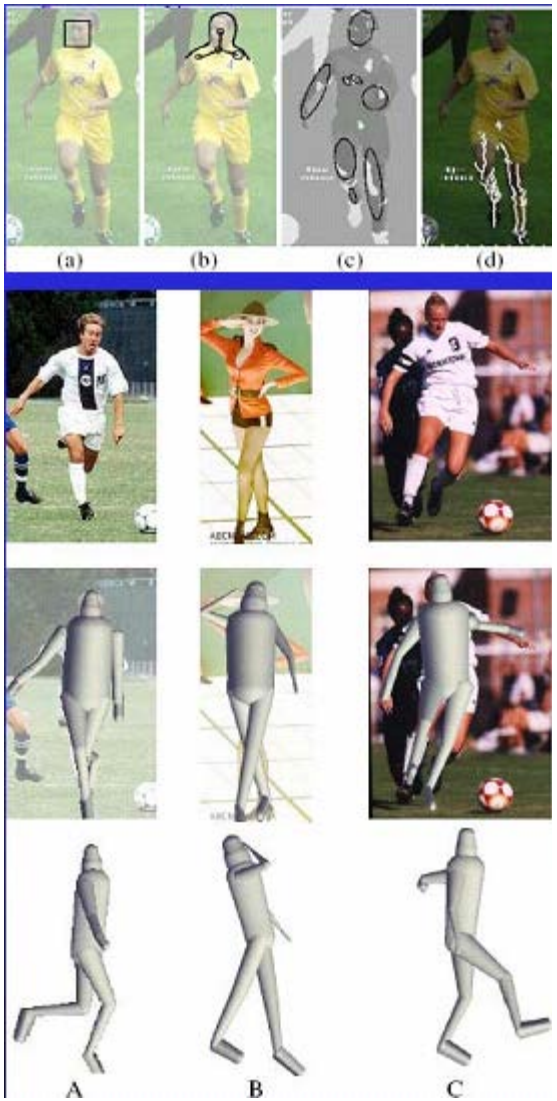


Figure 4: Example of detecting different sized faces

Figure 5: Detection result of a face with changes of expression

Utilitate

- Prima etapa in sistemele de recunoastere automata a feţelor
- Prima etapa in sistemele de “surveillance”
- Folosita ca ipoteza in identificarea persoanelor, a corpului uman, a partilor corpului uman
- Etapa de initializare in urmarirea (tracking-ul) fetelor sau a corpului in secvente de imagini
- Varietate mare de alte aplicatii



Dificultati

- Dimensionalitatea mare a spatiului se trasaturi:

Imagine grayscale (8 bits/pixel) de dim. 19 × 19

$\Rightarrow 256^{361} = 2^{8\times361} = 2^{2888}$ combinatii posibile ale intensitatilor

- Orientarea fetei (rotatia in afara planului imaginii)
  - Axa capului verticala: frontal, profil, semiprofil
  - Axa capului non-verticala
- Rotatie (in planul imaginii / in jurul axei optice a camerei)
- Conditii variabile de achizitie a imaginilor: iluminare, parametri camerei (senzor, castig, rezolutie imagine, obiectiv)
- Ocluzii
- Expresia faciala: fata = obiect non-rigid cu aparenta variabila
- Prezenta / absenta elementelor structurale: ochelari, barba, mustata etc.

Obiective

Probleme conexe

- **Localizarea fetei**

  - determinarea pozitiei unei singure fete intr-o imagine

- **Detectia elementelor faciale componente**

  - prezenta si locatia: ochi, sprancene, nas/nari, gura, buze, urechi etc.

- **Recunoasterea / identificarea fetelor**
- **Recunoasterea expresiei faciale**
- **Estimarea pozitiei corpului uman si urmarirea acestuia**

Teme de cercetare

- Reprezentarea fetei – cum se descrie o fata ?
- Scalarea – cum se rezolva problema scalei variabile ?
- Strategia de cautare – cum cautam o fata in imagine ?
- Viteza – cum marim viteza de procesare ?
- Precizia – localizare si rata de eroare a detectei (TPR, FPR, FNR)
- Postprocesare – combinarea rezultatelor detectiei

Metode

Metode de detcetie si localizare a feţei

1. Metode bazate pe cunostiinte:
   - ex: relatii spatiale intre trasaturile faciale pe baza caracteristicilor antropomorfice
2. Bazate pe trasaturi invariante (la pozitie, orientare, perspectiva)
3. Metode bazate pe potrivirea de sabloane (template matching)
   - sabloane pentru intreaga fata sau pentru anumite parti
4. Metode bazate pe aparente
   - modele / template-uri invatate pe un set de imagini de antrenare care sa surprinda variabilitatea aprentelor faciale

Abordari specifice sursei de imagini:

- Imagini statice sau secvente video
- Imagini grayscale/color

Metode

Resurse

http://www.facedetection.com/

- BD de antrenament
- aplicatii
- documentatii

Metode bazate pe culoare

Se bazeaza pe distriburtia de culoare specifica culorii pielii

- rasa/etnie
- conditii de iluminare

Spatii de culoare

- RGB, RGB normalizat, HSV, HIS, YCrCb, YIQ, UES, CIE, XYZ, CIE LIV

Analiza statistica

- histograma, LUT, model gausiene sau mixturi de modele gaussiene

Metoda 1

A FACE DETECTION TUTORIAL

In this tutorial, a simple face detection method is implemented by using Matlab 7.0.4. Several stages are involved in this method which includes skin detection, region analysis and template matching. The program implemented is not so good in terms of performance but hopefully this tutorial will give some basic idea of image processing for those who are new to this field.

Referinta si surse Matlab:

http://se.cs.ait.ac.th/cvwiki/matlab:tutorial:detectface

Sau

http://www-cs-students.stanford.edu/~robles/ee368/main.html

Metoda de detectie a feţei bazata pe culoare si potrivire de sabloane

Modelul de culoare YC_bC_r

1. Construirea unui model de culoare pentru piele

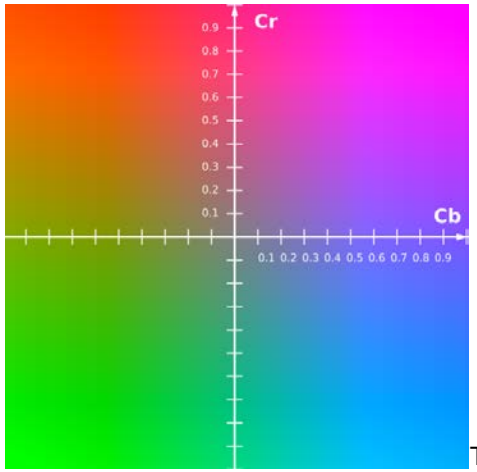
- Modele de culoare: rgb sau YCbCr

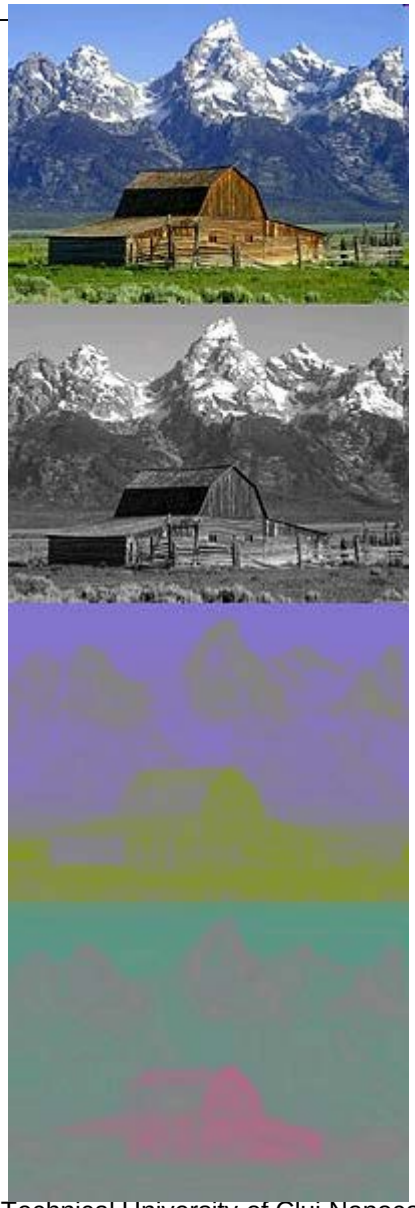
http://en.wikipedia.org/wiki/YCbCr

Y – luminanta; Cb, Cr – compomnente cromatice (diferenta albastru/ rosu)

$$\begin{pmatrix} Y \\ C_B \\ C_R \end{pmatrix} = \begin{pmatrix} 16 \\ 128 \\ 128 \end{pmatrix} + \frac{1}{256} \begin{pmatrix} 65.738 & 129.057 & 25.064 \\ -37.945 & -74.494 & 112.439 \\ 112.439 & -94.154 & -18.285 \end{pmatrix} \cdot \begin{pmatrix} R_N \\ G_N \\ B_N \end{pmatrix}$$

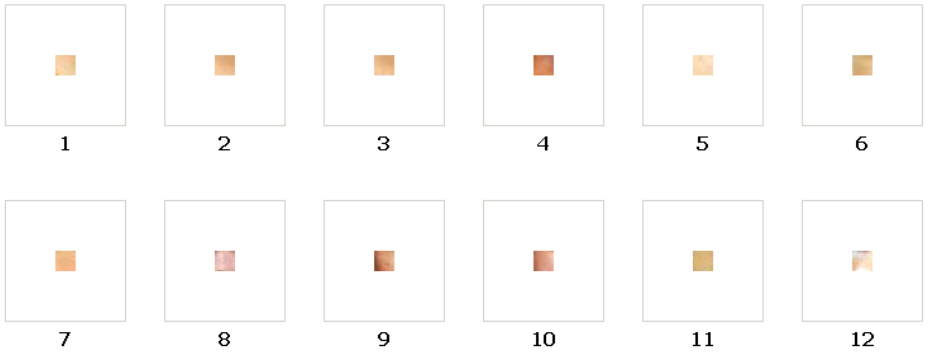
$$\begin{pmatrix} R_N \\ G_N \\ B_N \end{pmatrix} = \begin{pmatrix} 0.842 & 0.156 & 0.091 \\ -0.129 & 1.320 & -0.203 \\ 0.008 & -0.069 & 0.897 \end{pmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$





Modelul de culoare pentru piele

Set de imagini de test – se aleg zone de piele (ex: 16 x 16)



Histograma 2D pt. fiecare zona $\Rightarrow$ Histograma cumulativa (model gausian)

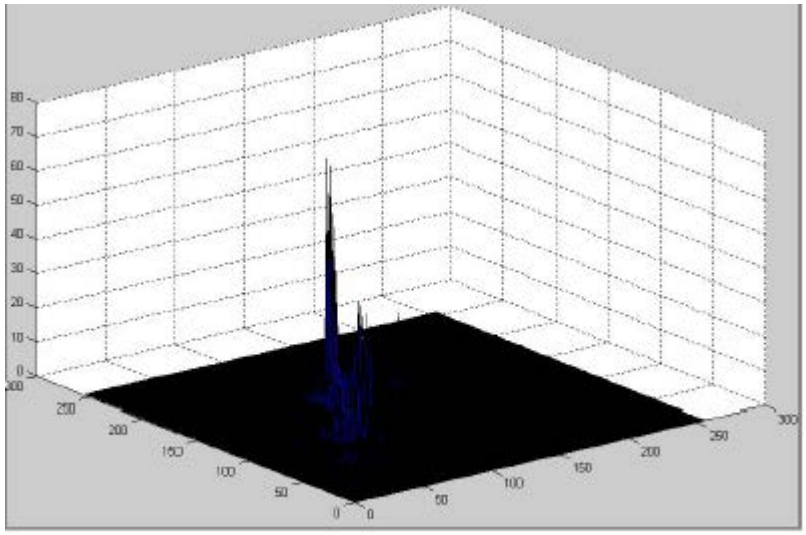


Fig. 1. 2D Histogram

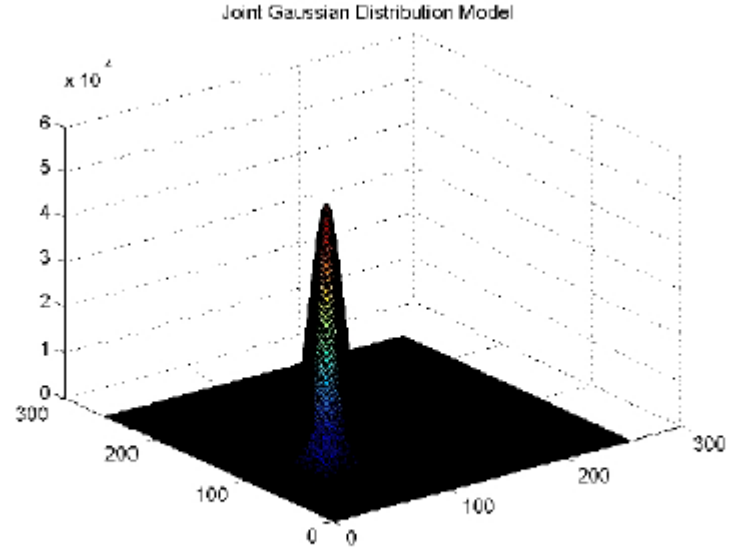


Fig. 2. Gaussian Skin Model

Modelul gausian (Fig. 2): medie si covarianta (CbCr)

Media si covarianta

Caracterizarea statistica a variabilelor aleatoare

1. Expectation: reprezinta centrul de masa al unei densitati de probabilitate.

$$E[X] = \mu = \int_{-\infty}^{+\infty} x f_X(x) dx$$

2. Varianta: reprezinta “imprastierea” in jurul mediei

$$VAR[X] = E[(X - E[X])^2] = \int_{-\infty}^{+\infty} (x-\mu)^2 f_X(x) dx \qquad STD[X] = VAR[X]^{1/2}$$

Caracterizarea statistica a vectorilor aleatori

Putem descrie partial un vector aleator prin urmatoarele valori:

1. Vectorul mediu:

$$E[X] = [E[X_1]E[X_2]...E[X_N]]^T = [\mu_1 \mu_2 ... \mu_N] = \mu$$

2. Matrticea de covarianta

$$COV[X] = \Sigma = E[(X-\mu)(X-\mu)^T]$$

$$= \begin{bmatrix} E[(x_1-\mu_1)(x_1-\mu_1)] & \dots & E[(x_1-\mu_1)(x_N-\mu_N)] \\ \dots & \dots & \\ E[(x_N-\mu_N)(x_1-\mu_1)] & \dots & E[(x_N-\mu_N)(x_N-\mu_N)] \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \dots & c_{1N} \\ \dots & \dots & \\ c_{1N} & \dots & \sigma_N^2 \end{bmatrix}$$

Matricea de covarianta

Matricea de covarianta indica tendinta fiecarei perechi de trasaturi (pozitii in vector) sa varieze impreuna, sau sa co-varieze. Covarianta are cateva proprietati importante:

- Daca $\mathbf{x_i}$ si $\mathbf{x_k}$ cresc impreuna, atunci $\mathbf{c_{ik}}>0$
- Daca $\mathbf{x_i}$ tinde sa descreasca atunci cand $\mathbf{x_k}$ creste, atunci $\mathbf{c_{ik}}<0$
- Daca $\mathbf{x_i}$ si $\mathbf{x_k}$ sunt necorelate, atunci $\mathbf{c_{ik}}=0$
- $|\mathbf{c_{ik}}|< \sigma_i\sigma_k$, unde i este deviatia standard a lui $\mathbf{x_i}$
- $\mathbf{c_{ii}} = VAR(\mathbf{x_i})$

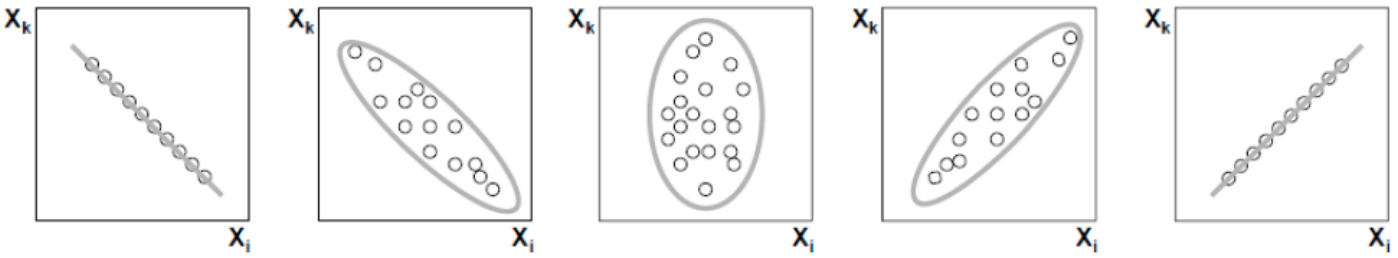
Termenii matricii de covarianta pot fi scrisi ca:

$$c_{ik} = E[(x_k - \mu_k)(x_i - \mu_i)]$$

$$c_{ii} = \sigma_i^2 \text{ and } c_{ik} = \rho_{ik}\sigma_i\sigma_k$$

unde ρ_{ik} este numit **coeficientul de corelatie**.

Exemple:



Metoda 1

2. Segmentarea zonelor

Fig. 3. Original Image

Fig. 4. Likelihood Image

Pentru fiecare pixel se calculeaza distanta (mahalanobis) la media distributiei gausiene antrenate $\Rightarrow$ normalizare (0 ..1) $\Rightarrow$ “skin likelihood” si reprezentare ca o imagine grayscale (Fig. 4)

Formally, the Mahalanobis distance of a multivariate vector $x = (x_1, x_2, x_3, \ldots, x_N)^T$ from a group of values with mean $\mu = (\mu_1, \mu_2, \mu_3, \ldots, \mu_N)^T$ and covariance matrix S is defined as:

$$D_M(x) = \sqrt{(x-\mu)^T S^{-1} (x-\mu)}.$$ [2]

Daca S (cov.) diagonala:

$$d(\vec{x}, \vec{\mu}) = \sqrt{\sum_{i=1}^{N} \frac{(x_i - \mu_i)^2)}{{\sigma_i}^2}}$$

Metoda 1

3. Binarizare adaptiva

Fig. 5. Skin Segmented Image

4. Operati morfologice

- umplere de regiuni, eroziune si dilatare, “and” logic cu img. pas 3

Fig. 6. Eroded Image

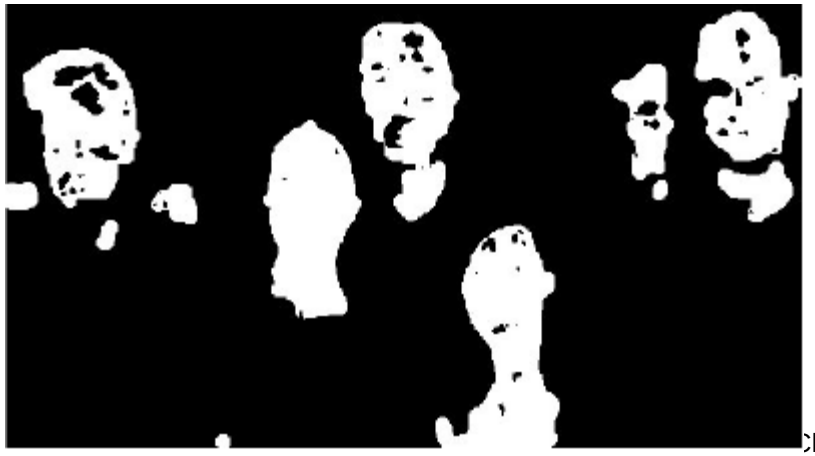


Fig. 7. Dilated Image

Metoda 1

5. Etichetare

Fig. 8. Colored Labeled Region

6. Calcul nr. Euler (fiecare eticheta) $\Rightarrow$ se retin doar regiuni cu nr. Goluri =1-nr_E >=1

Nr_E = total number of objects in the region minus the total number of holes

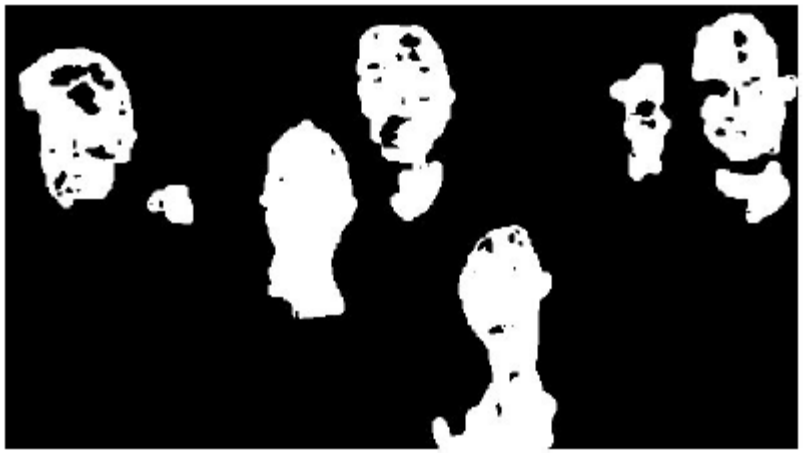


Fig. 9. After Euler Test

Metoda 1

7. Calcul factor de alungire (aspect ratio - AR)

- AR = lg. axa majora / lg. axa minora
- se retin obiecte cu AR = 1 … 3,5

Fig. 10. After Aspect Ration Test

Metoda 1

8. Template matching

- template: imaginea medie la N fete (grayscale)

- pt. fiecare regiune candidat se calculeaza dimensiunea si orientarea (axa de alungire)

- se compara (crosss-corelation) cu img. template-ului scalata si rotita corespunzator

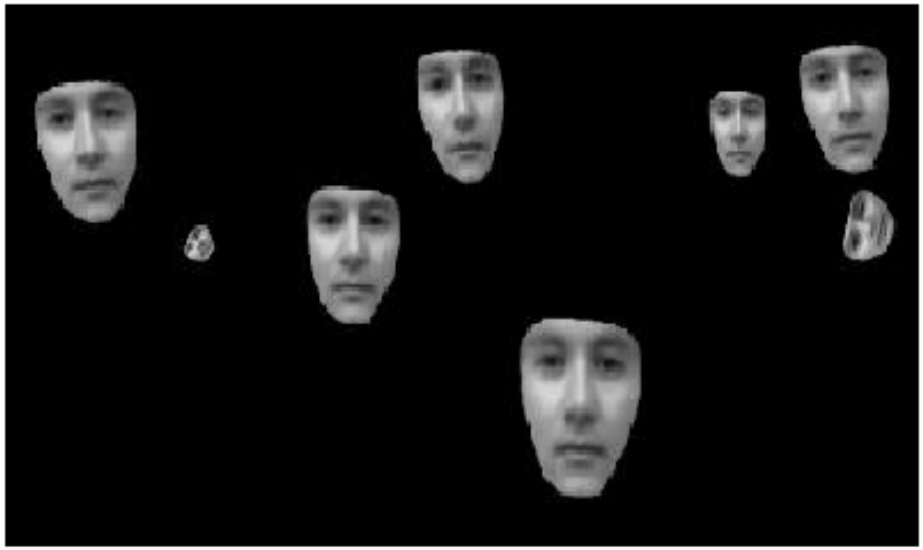


Fig. 11. Template Matching

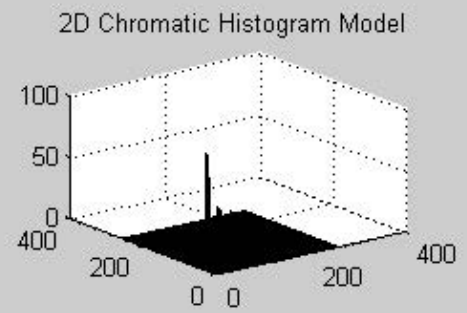
Metoda 1

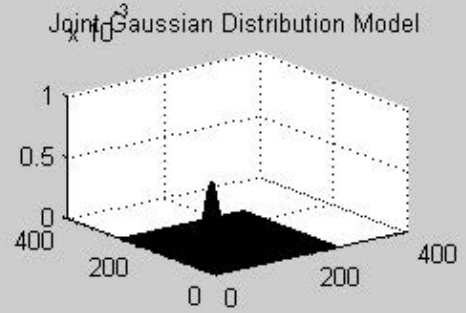
9. Rezultate

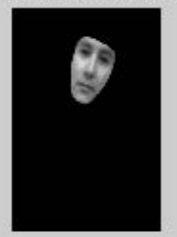
Fig. 12. Face Detection

Exemple rulare:

face_detection ('testset/blackgirl.jpg')

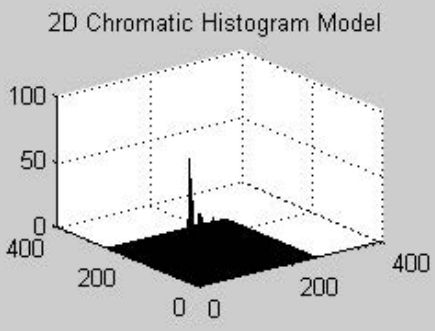


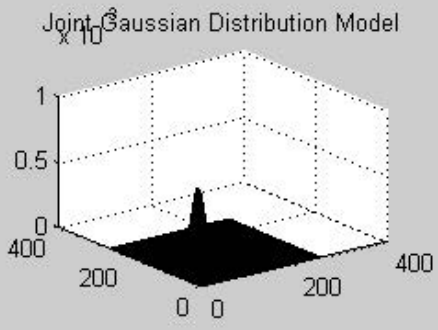


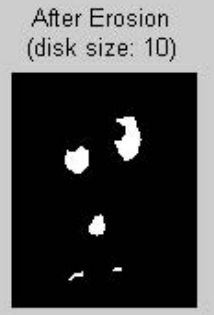


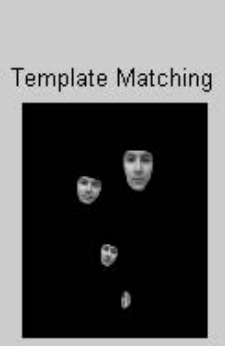
Exemple rulare:

face_detection ('testset/family.jpg')









IOC

Metoda 2

“A Real-Time Vision Interface Based on Gaze Detection — EyeKeys”, John J. Magee

Referinte

B. Kisacanin, V. Pavlovic, T.S. Huang, Real-Time Vision for Human-Computer Interaction, *Springer 2005*, pp. 141-157.

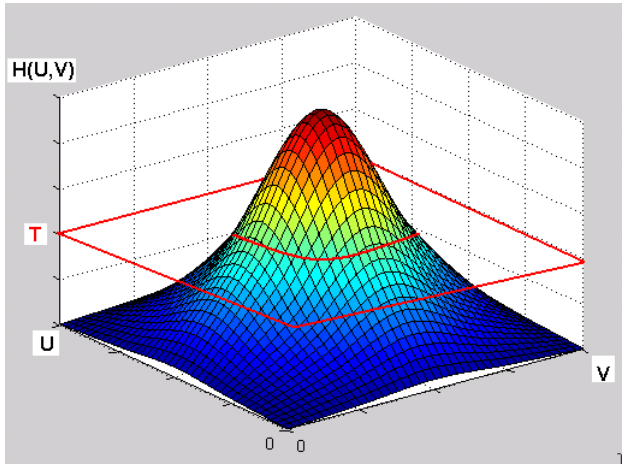
- **Metoda de detectie a feţei bazata pe culoare si potrivire de sabloane**

Metoda 2

Scop: gasirea unei masti pt. fata (zonei cu piele)

Spatiul de culoare YUV:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.437 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$



Gasirea mastii:

- Se considera un set de 15 imagini de antrenare (acceasi persoana, conditii similare de iluminare). La schimbarea conditiilor $\Rightarrow$ reantrenare

- Se selecteaza manual zone de pe fata (piele) si realizeaza histograma UV pt. aceste zone si apoi se face histograma cumulativa pt. 15 imagini.

-Se stabileste o valoare de prag T

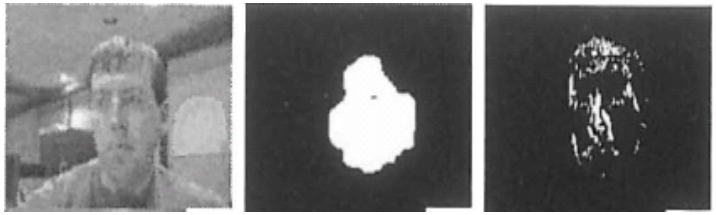
-Pentru fiecare pixel din imaginea curenta, daca valoarea (U,V) corespunde unui punct din histograma care depaseste valoarea T se marcheaza ca punct de fata/masca

-Masca va fi o imagine binara (alb – fata, negru – in rest)

Metoda 2

Diferente intre cadre succesive (camera fixa)

- Daca exista miscare ⇒ diferentele sunt evidentiate prin tresholding (in special in zonele cu gradient nenul) ⇒ masca a miscarii:



Masca culoare | Masca miscare

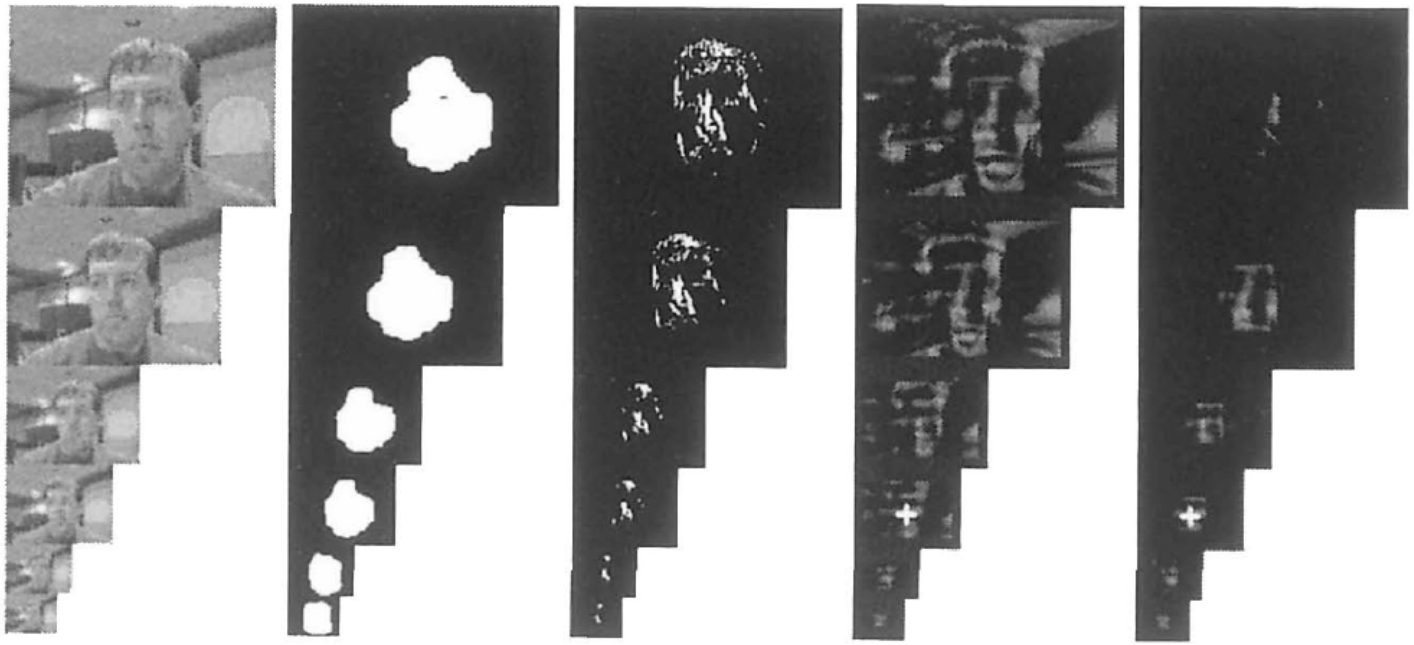
-Daca nu exista miscare ⇒ se foloseste ca masca zona detectata a fetei din imaginile precedente

Detectia fetelor

– potrivire de sabloane prin cautare piramidala

Metoda 2

Scalarea imaginilor si a sabloanelor pt. detectia fetei



P_Y P_{color_mask} P_{motion_mask} $P_{correlation}$ $P_{corelation} \cap P_{color\_mask} \cap P_{motion\_mask}$

Nivel	Latime	Inaltime	Factor de scalare
0	640	480	1
1	320	240	2
2	160	120	4
3	128	96	5
4	80	60	8
5	64	48	10
6	40	30	16
7	32	24	20

Sablon:

- dim. fixa (12x16)
- medie a intensitatii (Y) pt. un set de antrenarte de 8 fete

Metoda 2

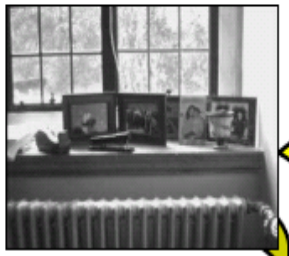
Generarea unei piramide gaussiene (factor de decimare 2)

Convolutie cu gaussian de dimensiune fixa

Original Image
(Level 0)

$$G_l(i,j) \sum_m \sum_n w(m,n)\, G_{l-1}(2i+m, 2j+n)$$

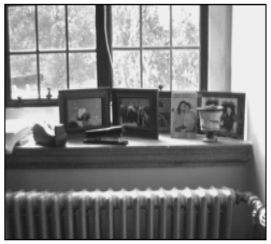
1	4	6	4	1
4	16	24	16	4
6	24	36	24	6
4	16	24	16	4
1	4	6	4	1



Decimation
(Level 1)

Decimation
(Level 2)

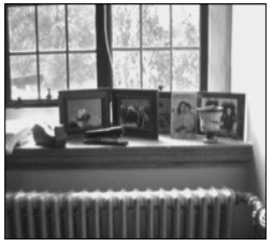
Decimation
(Level 3)



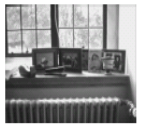
*

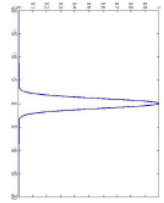
1	4	6	4	1
4	16	24	16	4
6	24	36	24	6
4	16	24	16	4
1	4	6	4	1

Filter

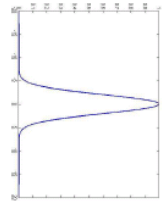


Subsample

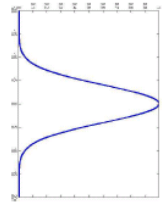




Level 1



Level 2

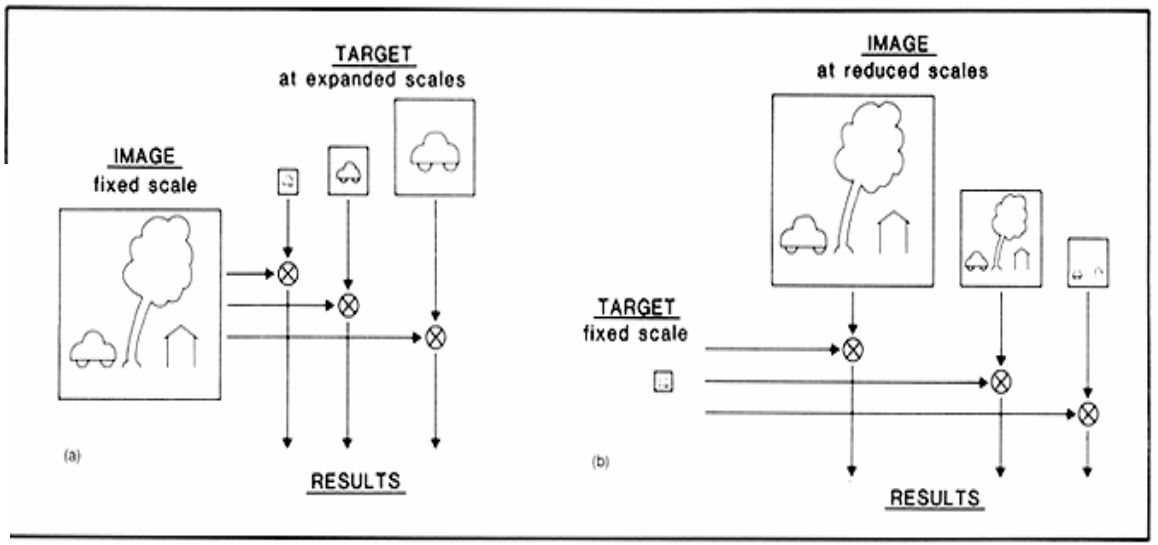


Level 3

Metoda 2

Scalarea imaginilor si a sabloanelor

Template



Search Region

Original Image

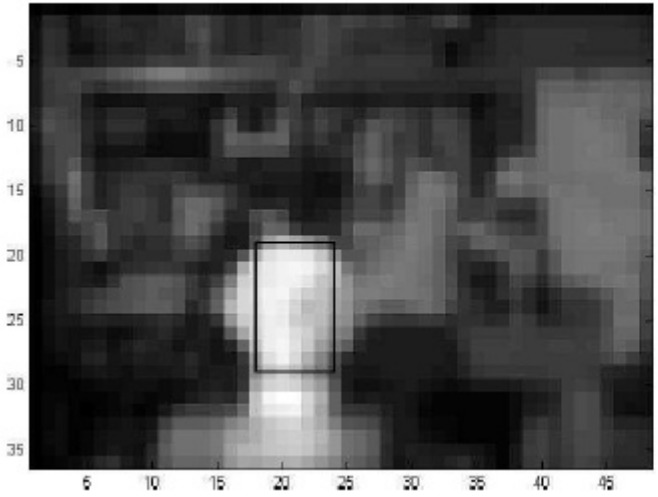
Metrica de corelatie

- SAD, SSD etc

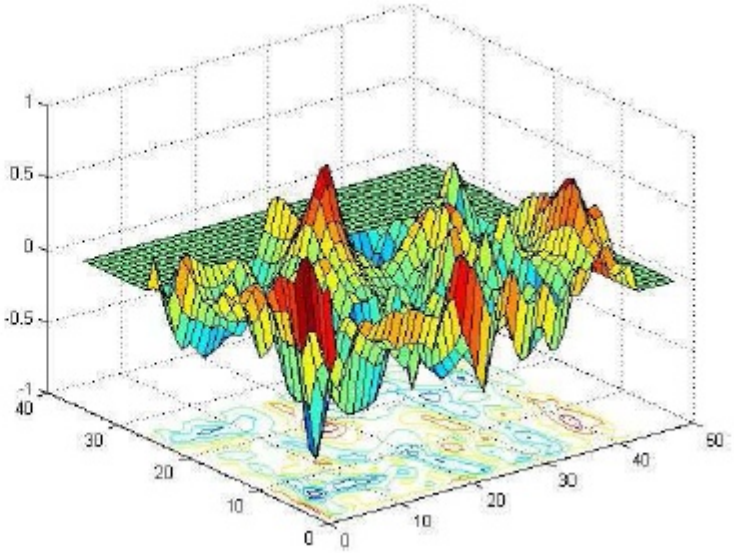
$$SAD(x, y_R) = \sum_{i=-\frac{w}{2}}^{\frac{w}{2}} \sum_{j=-\frac{w}{2}}^{\frac{w}{2}} \left| I(x + i, y + j) - T(i, j) \right|$$

Metoda 2

- Cautarea incepe de la nivelul cel mai de jos al piramidei (ex. Level 3)
- La acest nivel se inceraca corelarea templetului corespunzator peste intreaga imagine



Pozitia (xm, ym) din imagine corespunzatoare Min(SAD)

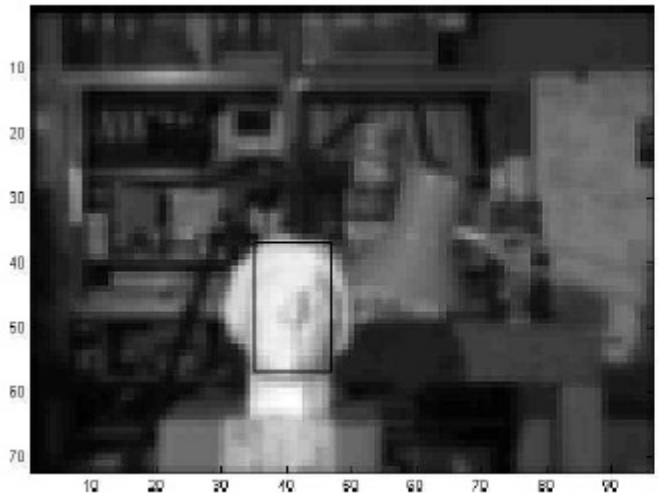


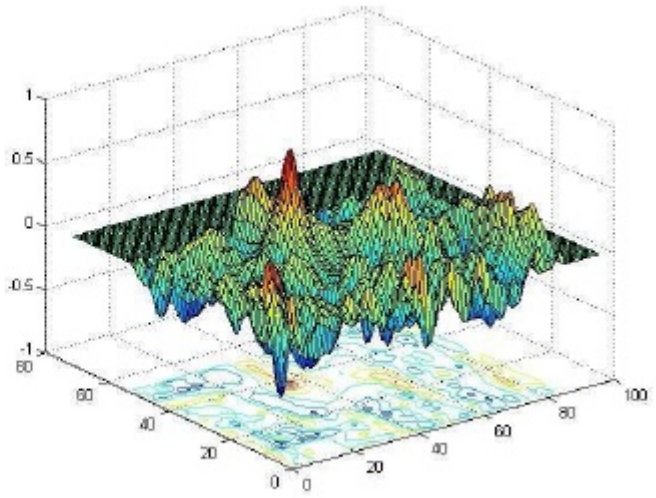
SAD

Metoda 2

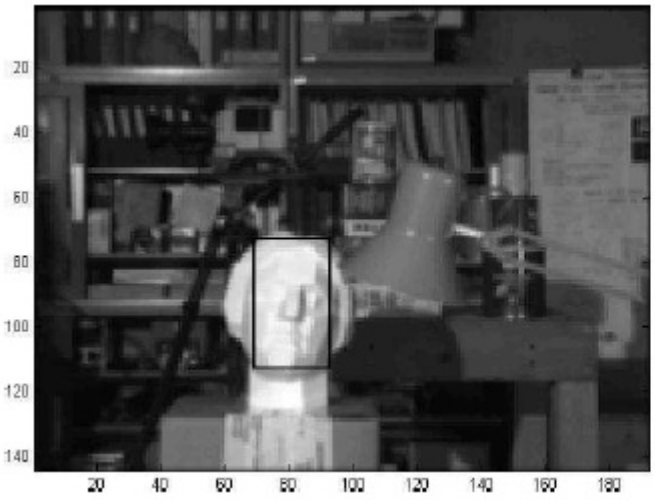
Cautarea in nivelul urmator (ex. Level 2) se face doar intr-o vecinatate a pozitiei minimului de la nivelul de mai jos:

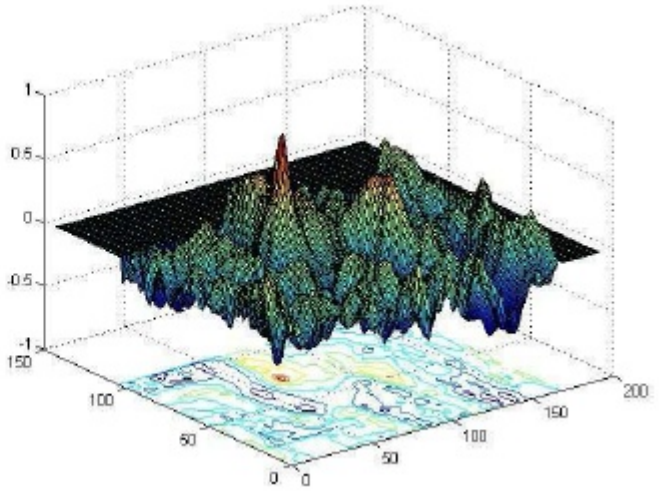
Nivel 2





Nivel 1





Detectia fetelor in OpenCV

Resurse

http://sourceforge.net/projects/opencvlibrary/
http://opencv.willowgarage.com/wiki/FaceDetection
http://nashruddin.com/OpenCV_Face_Detection
http://nashruddin.com/OpenCV_Eye_Detection
http://www.mathworks.com/matlabcentral/fileexchange/19912 (Matlab)
http://www.cs.princeton.edu/courses/archive/fall08/cos429/CourseMaterials/Precept1/facedetect.pdf
http://note.sonots.com/SciSoftware/haartraining.html (trainning)

Ming-Hsuan Yang,David J. Kriegman, Narendra Ahuja, Detecting Faces in Images: A Survey, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 24, NO. 1, JANUARY 2002

Demo

http://vimeo.com/12774628 (Haar cascade classifier demo)

Bibliografie

http://research.microsoft.com/en-us/um/people/viola/Pubs/Detect/violaJones_IJCV.pdf
http://cs.nyu.edu/~eugenew/publications/viola-facedet04-talk.pdf

Metoda Viola & Jones

Etape/caracteristici:

1.Extragere trasaturi rectangulare

2.Clasificare prin “ boosting”

3.Algortm de detectie multi-rezolutie

Trasaturi:
- rectangulare (grayscale) $\approx$ Haar
- “imagine integrala” (marirea vitezei de calcul)

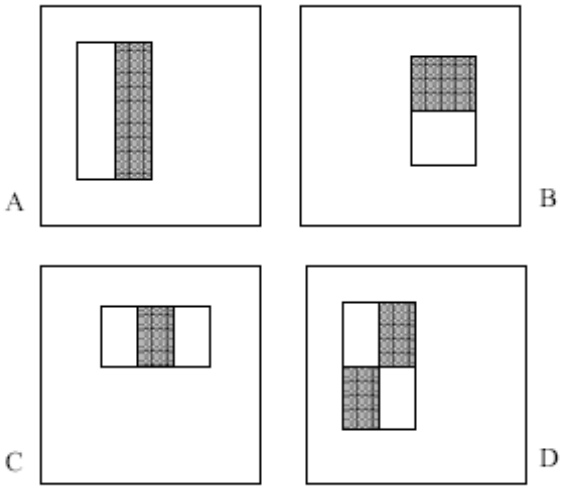
Clasificare:
- trasaturile $\Rightarrow$ clasificatori slabi
- cascada de clasificatori slabi (concatenare seriala)
- “boosting” anumite trasaturi (AdaBoost) $\Rightarrow$ detector suficient de bun

Viteza: 700Mhz Pentium III, 384x288 $\approx$ 0.067 sec.

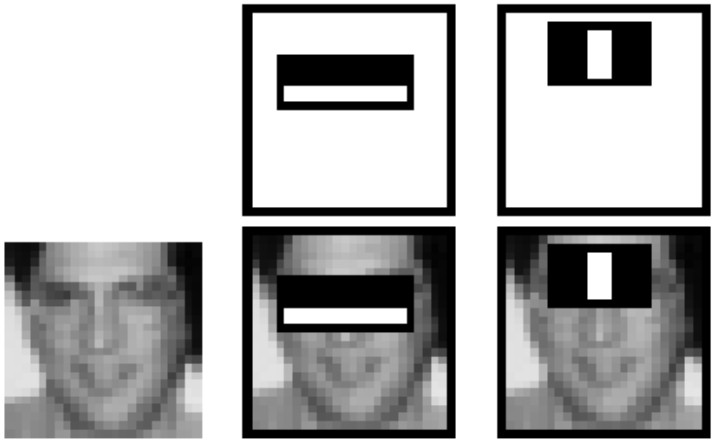
Metoda Viola & Jones

Trasaturi rectangulare

- 4 tipuri de masti generale:



Pt. detectia de fete:



Calcul:

- Se suprapun mastile in diverse pozitii peste imaginea grayscale
- Pt. fiecare pozitie/sablon se calculeaza:

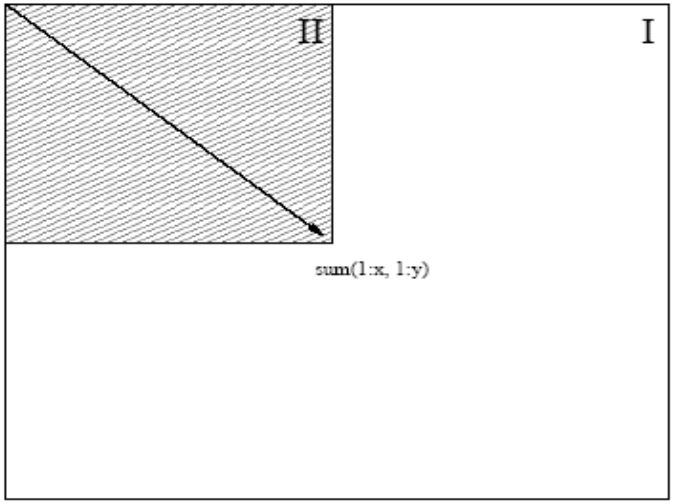
$$Value = \sum (intensity\ in\ white\ area) - \sum (intensity\ in\ black\ area)$$

- La scala minima pt o fata (24 x 24 pixeli) $\Rightarrow$ set trasaturi rectangulare ~160,000 !!!

Metoda Viola & Jones

Optimizare calcul trasaturi rectangulare

- Imagine INTEGRALA (a imainii sursa / de intrare)



$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x',y')$$

Imaginea integrala se poate obtine prin parcurgerea (o singura data) a imagini initiale

$$s(x,y) = s(x,y-1) + i(x,y)$$

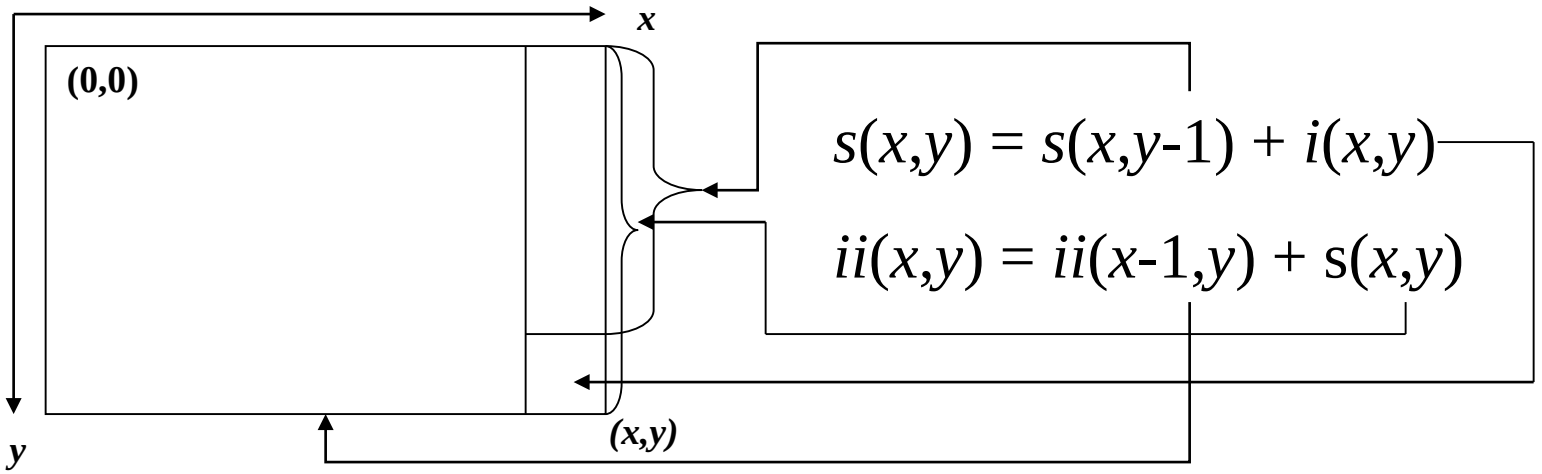
$$ii(x,y) = ii(x-1,y) + s(x,y)$$

$$s(x,-1) = 0$$

$$ii(-1,y) = 0$$

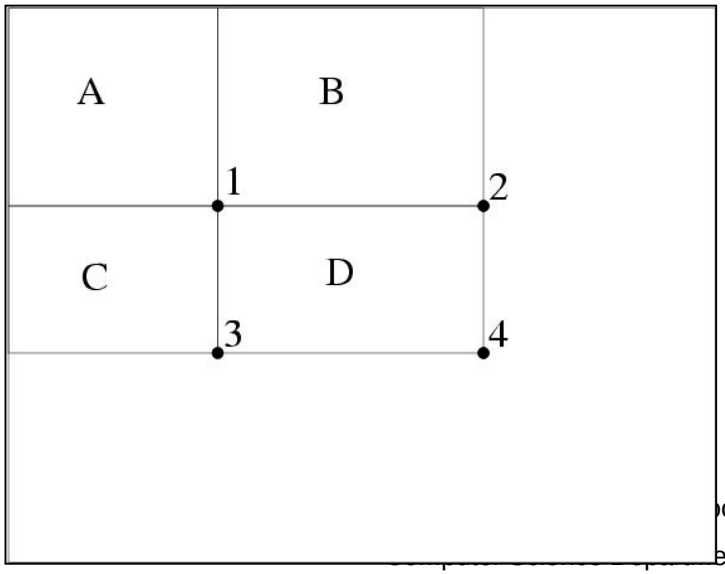
Metoda Viola & Jones

Calculul imaginii INTEGRALE



Calculul intensitatii pixelilor din deptunghiul D (se cunosc A, A+B si A+C = imaginile integrale din pixelii 1, 2 respectiv 3):

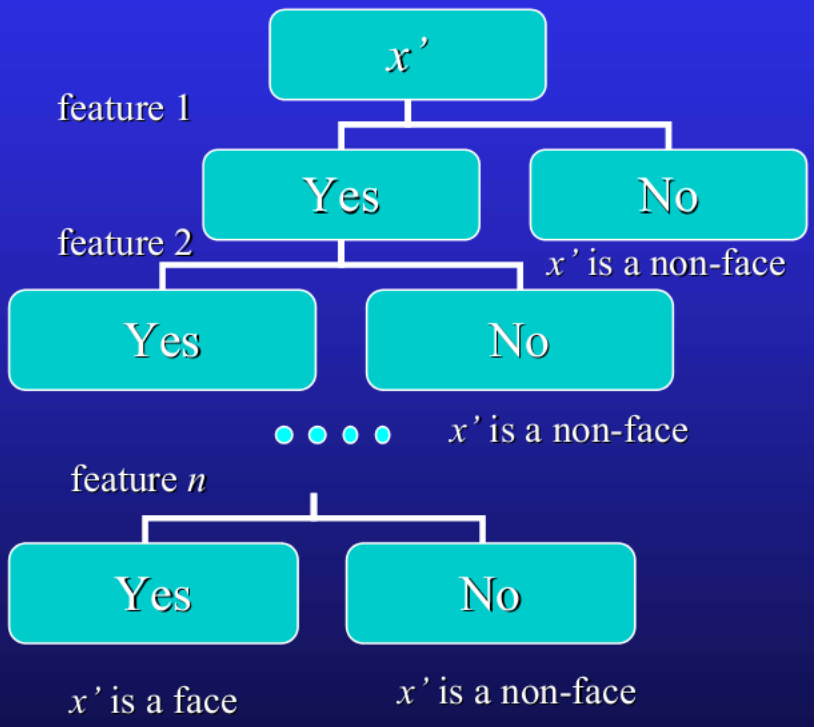
$D = ii(4) - (A+B+C)$
$= ii(4) - \{ii(1) + [ii(2)-ii(1)] + [ii(3)-ii(1)]\}$
$= ii(4) + ii(1) - ii(2) - ii(3)$



Metoda Viola & Jones

Feature Selection [Viola and Jones 01]

- Training: If *x* is a face, then *x*
  - most likely has feature 1 (easiest feature, and of greatest importance)
  - very likely to have feature 2 (easy feature)
  - …
  - likely to have feature n (more complex feature, and of less importance since it does not exist in all the faces in the training set)
- Testing: Given a test sub-image *x'*
  - if *x'* has feature 1:
    - Test whether *x'* has feature 2
      - Test whether *x'* has feature n
        - …
      - else …
    - else, it is not face
  - else, it is not a face
- Similar to decision tree

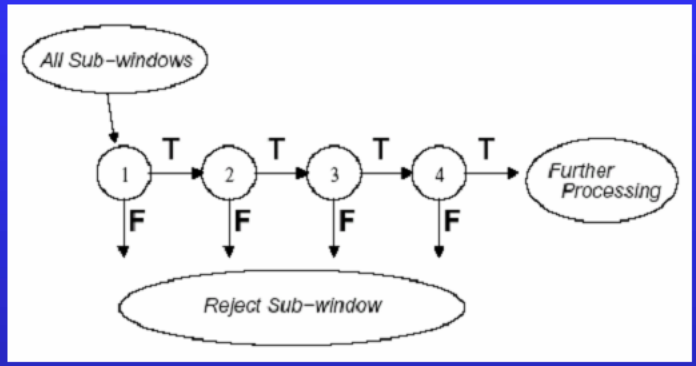


One simple implementation

Metoda Viola & Jones

Training Attentional Cascade

- Similar to decision tree
- Design parameters:
  - Number of cascade stages
  - Number of features of each stage
  - Threshold of each stage
- Example: 32 stage cascade classifier
  - 2-feature classifier in the firs stage → rejecting 60% non-faces while detecting 100% faces
  - 5-feature classifier in the second stage → rejecting 80% non-faces while detecting 100 % faces
  - 20-feature classifier in stages 3, 4, and 5
  - 50-feature classifier in stages 6 and 7
  - 100-feature classifier in stages 8 to 12
  - 200-feature classifier in stage 13 to 32



[Viola and Jones 01]

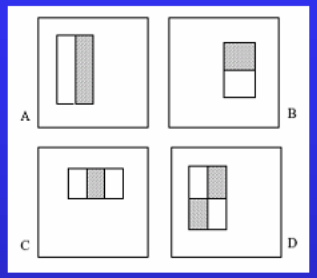
Metoda Viola & Jones

Boxlet As Weak Classifier [Viola & Jones 01]

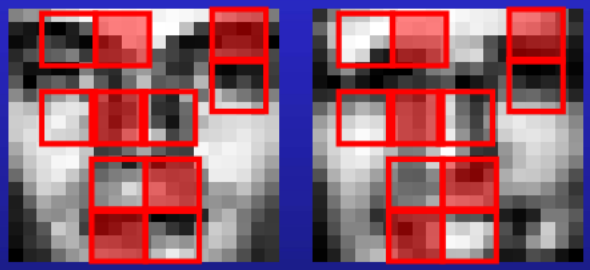
- Boxlet: compute the difference between the sums of pixels within two rectangular regions
  - Compute boxlets all over a pattern
  - Harr-like wavelets
  - Over-complete representation: lots of boxlet features
- For each boxlet j, compute $f_j(x)$ where x is a positive or negative example
- Each feature is used as a weak classifier
- Set threshold θ_j so that *most* samples are classified correctly:

$$h_j(x, f, p, \theta) = 1 \text{ if } f_j(x) < \theta_j \text{ (or } f_j(x) > \theta_j)$$

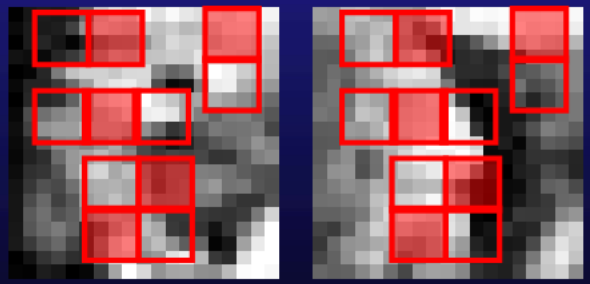
- Sequentially select the boxlets



Boxlet:
2-rectangle,
3-rectangle,
4-rectangle



face samples



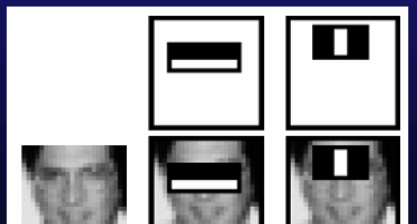
non-face samples

Selecting Features Using Adaboost

[Viola and Jones 01]

- For t=1, …, T
  - Construct a weak classifier using one single feature h_t $h_j(x)=\begin{cases}1 & \text{if } p_j f_j(x) < p_j\theta_j \\ 0 & \text{otherwise}\end{cases}$ where p_j is a parity bit and θ_j is a threshold
  - For each feature j, train a classifier h_j, the error is evaluated with respect to w_t, $\varepsilon_t=\Sigma_i w_i|h_j(x_i)-y_i|$
  - Choose the classifier h_t, with the minimum error ε_t
  - Update the weights: $w_{t+1,i}=w_{t,i}\,\beta_t^{\,1-e_i}$, $e_i=0$ if x_i is correctly classified, and $e_i=1$ otherwise. $\beta_t = \varepsilon_t/(1-\varepsilon_t)$
- Final classifier:

$$h_j(x)=\begin{cases}1 & \sum_{t=1}^{T}\alpha_t h_t(x) \geq \frac{1}{2}\sum_{t=1}^{T}\alpha_t \\ 0 & \text{otherwise}\end{cases} \quad ,\alpha_t=\log\frac{1}{\beta_t}$$



The top two boxlets selected by Adaboost

Metoda Viola & Jones

Rezultate si demo

- pt. raportarea unei fete detectate este necesara detectia de mai multe instante/detectii pozitive intr-o regiune

- pozitia finala a fetei $\Rightarrow$ non-maxima supression pe rezultatul clasificarii pe fiecare instana/detectie pozitiva

http://vimeo.com/12774628 (demo)

Imbunatatiri / extensii:

- Rotatii: in plan si in afara planului (“multi-pose”)
- Ocluzii
- Componente faciale si parti coprorale (OpenCV)

Metoda Viola & Jones - OpenCV

Clasifcatori HAAR in OpenCV pt. detectia fetei, componentelor faciale sau parti corporale:
%OPENCV_DIR%*data**haarcascades*\

haarcascade_eye.xml
haarcascade_eye_tree_eyeglasses.xml
haarcascade_frontalface_alt.xml
haarcascade_frontalface_alt_tree.xml
haarcascade_frontalface_alt2.xml
haarcascade_frontalface_default.xml
haarcascade_fullbody.xml
haarcascade_lefteye_2splits.xml
haarcascade_lowerbody.xml
haarcascade_mcs_eyepair_big.xml
haarcascade_mcs_eyepair_small.xml
haarcascade_mcs_leftear.xml
haarcascade_mcs_lefteye.xml
haarcascade_mcs_mouth.xml
haarcascade_mcs_nose.xml
haarcascade_mcs_rightear.xml
haarcascade_mcs_righteye.xml
haarcascade_mcs_upperbody.xml
haarcascade_profileface.xml
haarcascade_righteye_2splits.xml
haarcascade_upperbody.xml

Alti clasificatori pt. detectia de obiecte cu ajutorul metodei detectMultiScale ():

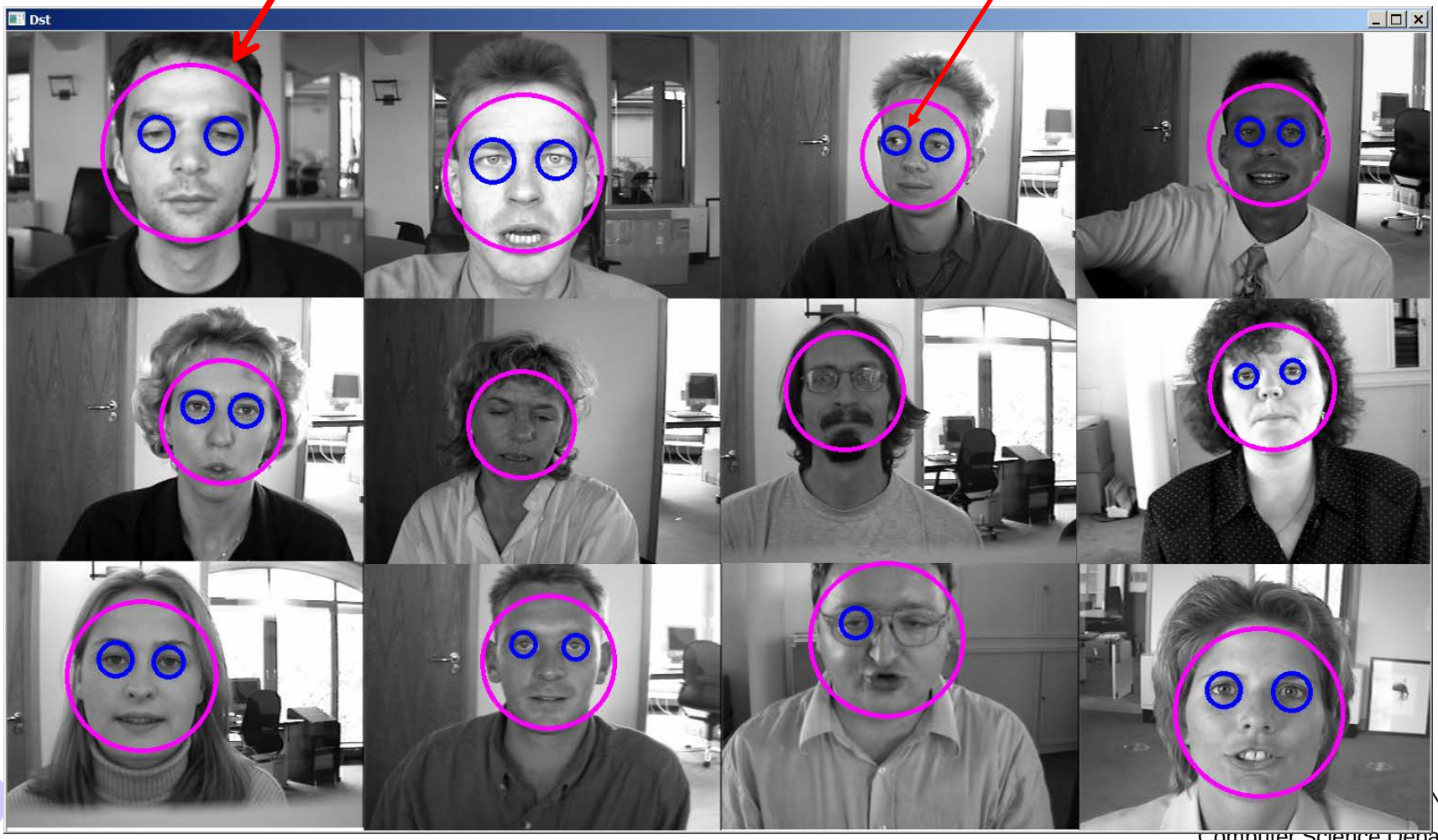
%OPENCV_DIR%\data\lbpcascades :
lbpcascade_frontalface.xml
http://en.wikipedia.org/wiki/Local_binary_patterns

%OPENCV_DIR%\data\hogcascades
hogcascade_pedestrians.xml
http://en.wikipedia.org/wiki/Histogram_of_oriented_gradients

Ex. OpenCV: detectMultiScale()

Sursa imagini: BioID Face Database
(http://www.bioid.com/downloads/software/bioid-face-database.html)

haarcascade_frontalface_alt.xml haarcascade_eye_tree_eyeglasses.xml



Detectia fetelor – deep learning

Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, *23*(10), 1499-1503. (2000 citari / Google Scholar) https://arxiv.org/ftp/arxiv/papers/1604/1604.02878.pdf
Implementari: https://machinelearningmastery.com/how-to-perform-face-detection-with-classical-and-deep-learning-methods-in-python-with-keras/

CNN cascadat (3 stagii) localizare si aliniere:

1. Generare ferestre candidat (retea cu adancime mica)
2. Eliminare candidati non-fata cu o retea mai complexa
3. Rafinare resultate si detectie puncte cheie

Performante timp real [Zhang2016]:

Method	GPU	Speed
Ours	Nvidia Titan Black	99 FPS
Cascade CNN [19]	Nvidia Titan Black	100 FPS
Faceness [11]	Nvidia Titan Black	20 FPS
DP2MFD [27]	Nvidia Tesla K20	0.285 FPS

Detectia fetelor – deep learning

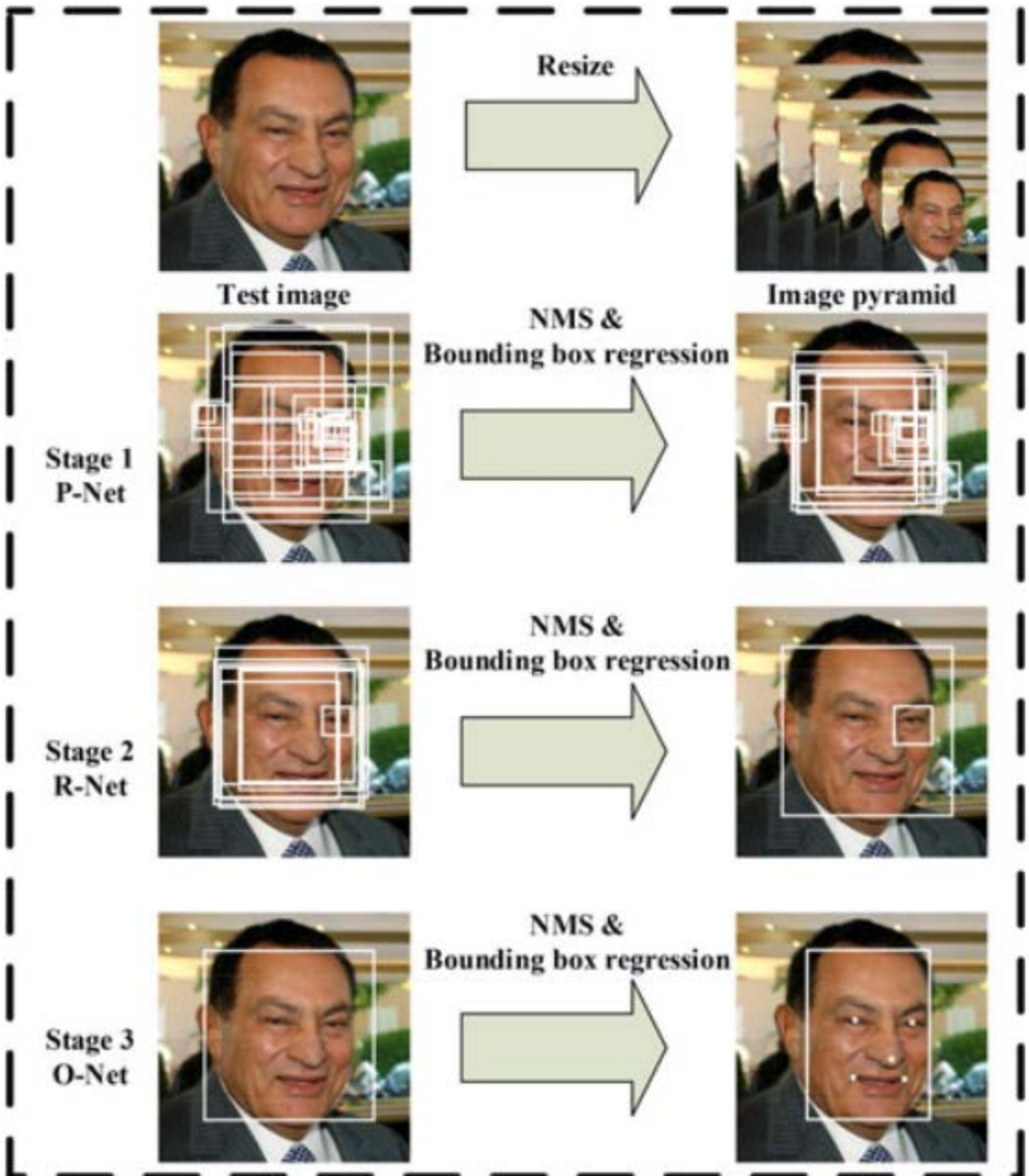
[Zhang2016]:

S0: Creare piramida de imagini

S1: Proposal network ⇒ ferestre candidat + NMS (non-maxima suppression

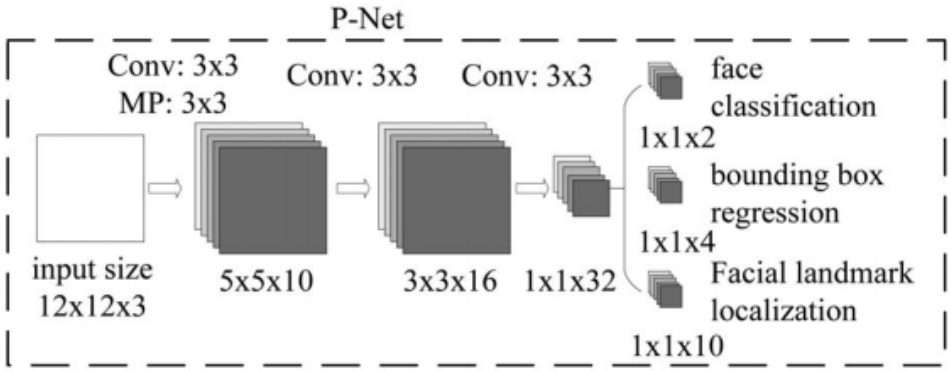
S2: Refine network ⇒ eliminare candidati falsi (“non-face” + NMS

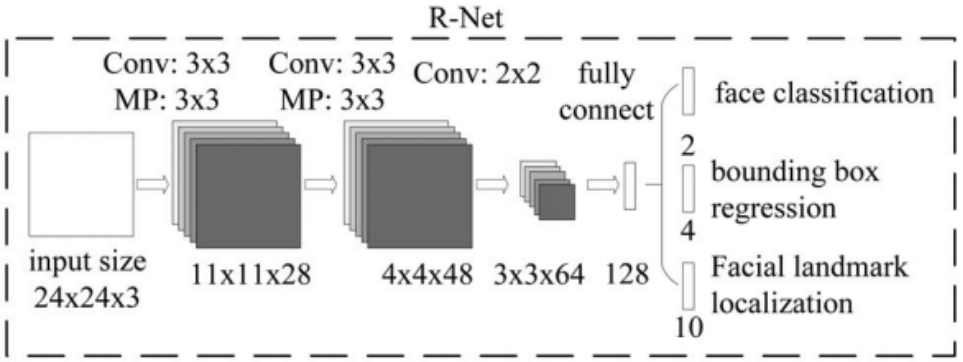
S3: Output network ⇒ localizare fata si 5 puncte cheie

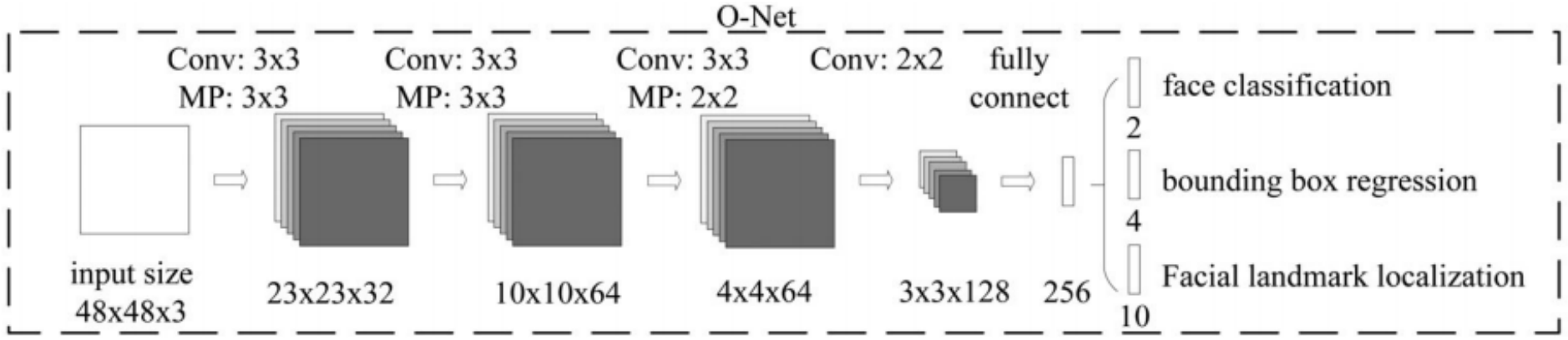


Detectia fetelor – deep learning

Arhitectura retelelor [Zhang2016]:







Detectia fetelor – deep learning

Antrenare [Zhang2016]:

1. Clasificatorul de fete (binary: fata/non-fata)
Cross entropy loss:

$$L_i^{\text{det}} = -\left(y_i^{\text{det}} \log\left(p_i\right) + \left(1 - y_i^{\text{det}}\right)\left(1 - \log(p_i)\right)\right)$$

y_i^{det} = *{1,0}* – GT label, p_i – probabilitatea ca instant x_i sa fie o fata

2. Regresia regiunilor de interes (ROI): offset-ul pana la cea mai apropiata GT ROI (ROI: left, top, height, width)
Euclidian loss:

$$L_i^{\text{box}} = \left\|\hat{y}_i^{\text{box}} - y_i^{\text{box}}\right\|_2^2$$

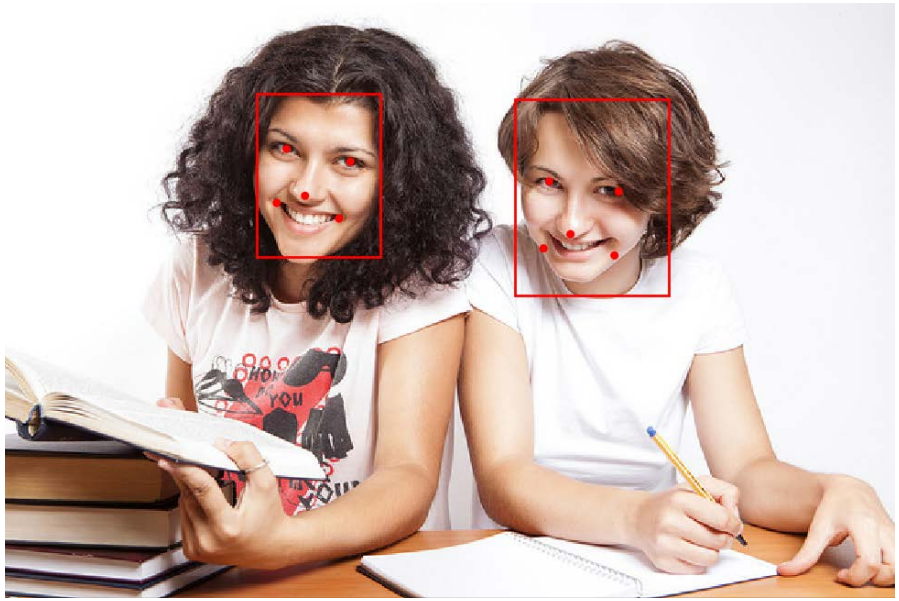
y_i^{box} = *{left,top,height, width}* GT, $y\hat{}_i^{box}$ = *{left,top,height, width} pred.*

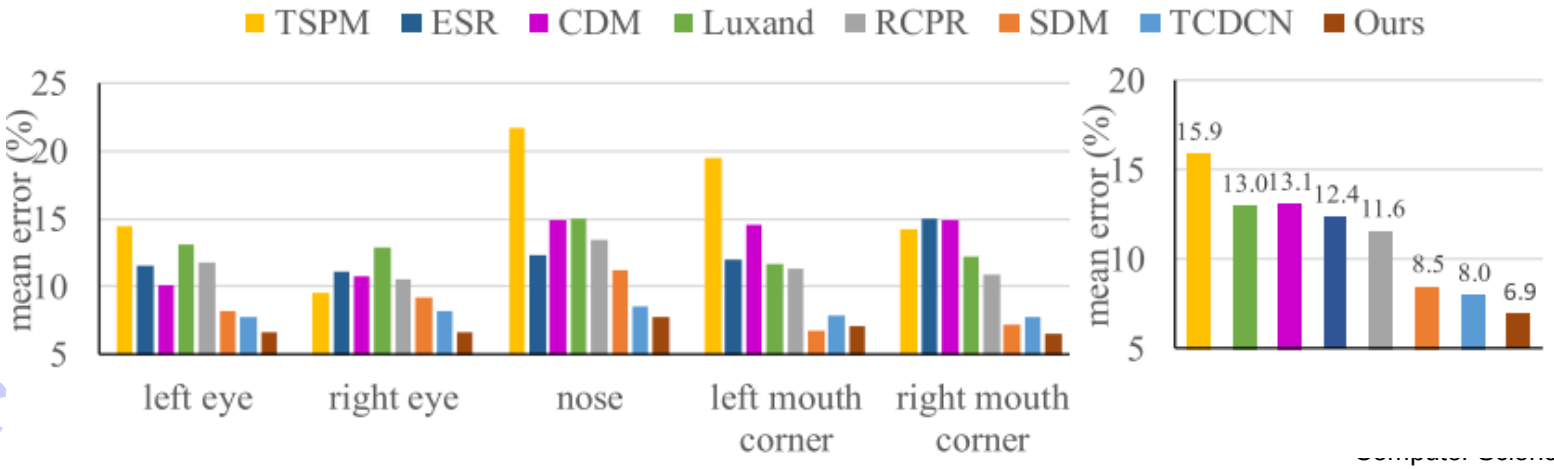
2. Localizarea punctelor cheie faciale:
Euclidian loss: $L_i^{\text{landmark}} = \left\|\hat{y}_i^{\text{landmark}} - y_i^{\text{landmark}}\right\|_2^2$

$y_i^{landmark}$ = *{(x,y)* | *left* eye, right eye, nose, left mouth corner, right mouth corner} GT ; $y\hat{}_i^{boxlandmark}$ *predictie*

Detectia fetelor – deep learning

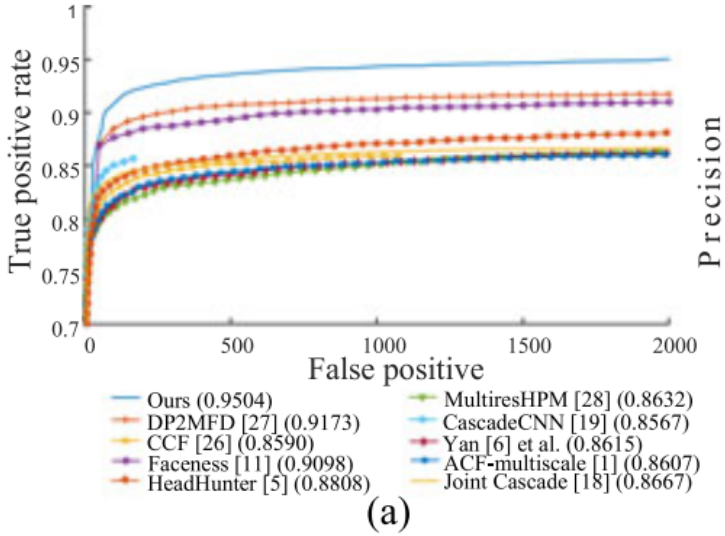
Rezultate



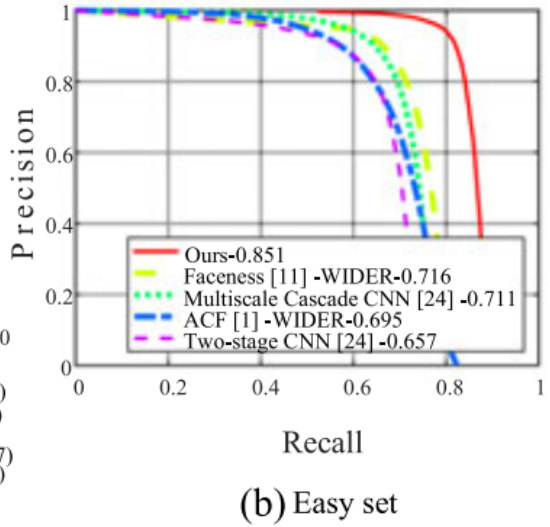


Detectia fetelor – deep learning

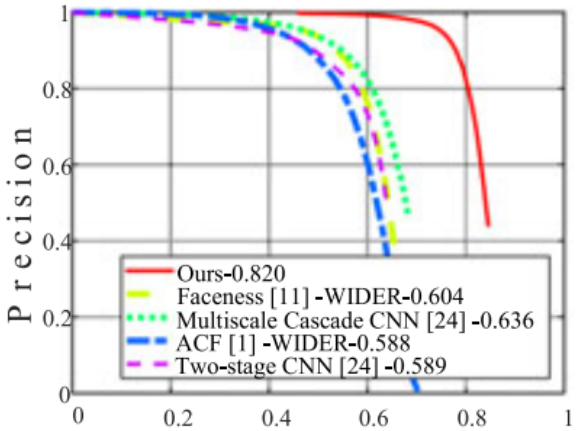
Rezultate [Zhang2016] - Data sets: face detection dataset and benchmark (FDDB), WIDER FACE, and annotated facial landmarks in the wild (AFLW)



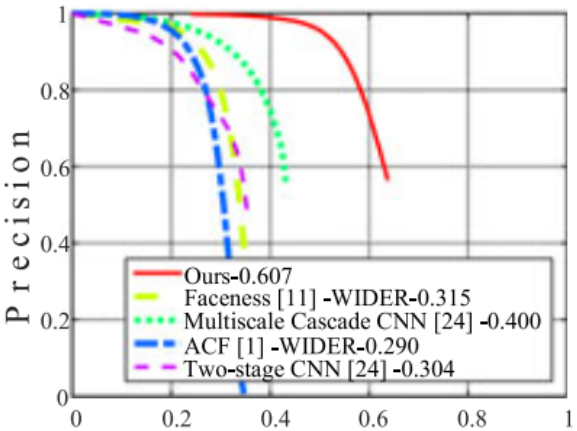
(a)



(b) Easy set



(c) Medium set



(d) Hard set

Detectia ochilor si urmarirea directiei privirii

Detectia de compnente faciale

Detectia ochilor si urmarirea directiei privirii

Cerinte:
hardware minimal (Web-cam +PC)
resurse de calcul minimale (timp de procesare)
Aplicatii:
- interfata pentru persoane cu dizabiltati/paralizie

Referinte
B. Kisacanin, V. Pavlovic, T.S. Huang, Real-Time Vision for Human-Computer Interaction, *Springer 2005*, pp. 141-157, “A Real-Time Vision Interface Based on Gaze Detection — EyeKeys”, John J. Magee

Detectia de compnente faciale

Detectia ochilor si urmarirea directiei privirii

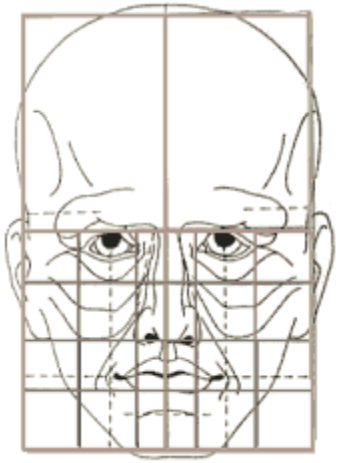
Metode alternative:

- Camere montate pe cap sau ochelari care urmaresc miscarea iris-ului $\Rightarrow$ simplifica detectia
- Electrozi montati pe fata (in jurul ochilor) care detecteaza impulsurile nervoase ale muschilor globilor oculari
- Iluminare cu lumina IR intermitenta (imagini diferenta intre iluminare normala si iluminare IR) $\Rightarrow$ simplifica detectia

Dezavantaje metode alternative

- Invazive
- Hardware specializat / costistor
- Procedura de calibrare a parametrilor elaborata

Analiza directiei privirii



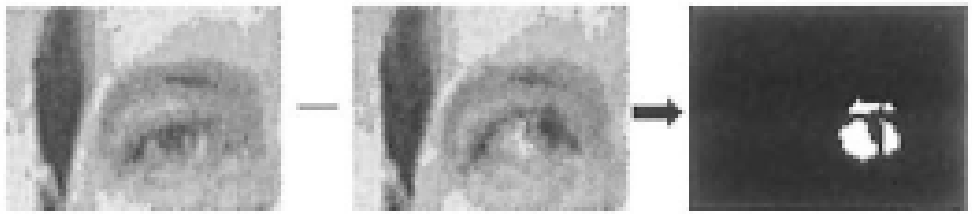
1. Obtinerea celor 2 imagini ale ochilor pe baza trasaturilor antopomorfice (regiunea ochilor trebuie să aibă înalţimea 1/8 din înălţimea totală a feţei şi laţimea 1/5 din laţimea totală a feţei) $\Rightarrow$ 2 sub-imagini care conţin ochii din imaginea cu rezoluţia cea mai mare (nivel 0).

- Dimensiunea subimaginilor depinde de scala la care s-a gasit faţa $\Rightarrow$ scalare la dimensiune fixa (60x80) pixeli prin interpolare bi-liniară.

2. Stabilizarea imaginilor ochilor (detectia si tracking-ul fetei nu este suficient de precis pt. miscari ale capului de cativa pixeli)

- Diferenţa dintre cadre succesive pentru a crea imagini binare de mişcare (background subtraction)

- Calculul momentelor de ordin 1 (centru de masa). Aceste puncte centroid sunt folosite pentru a estima locaţia ochilor in imaginea feţei (precizie buna pt. imagini de rezolutie mica)



Analiza directiei privirii

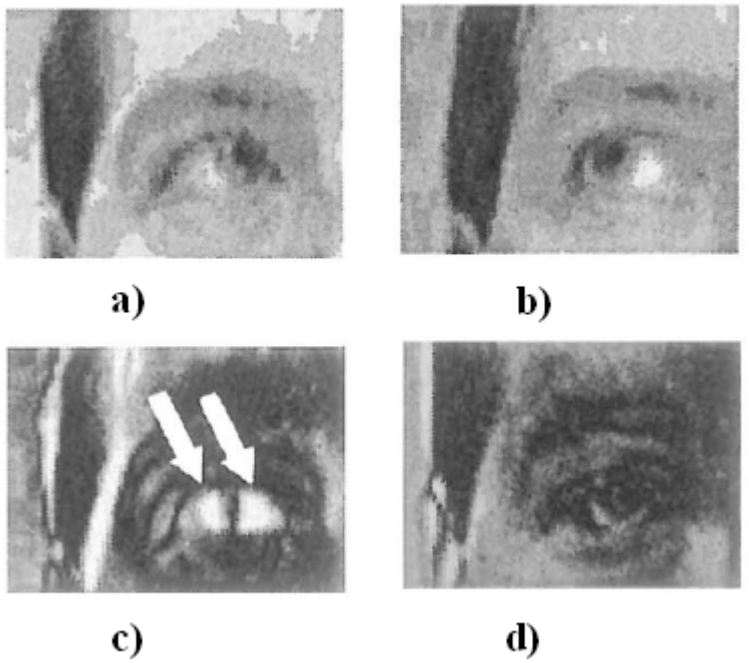
3. Comparatie intre ochiul stang si drept

Ochiul stâng si ochiul drept sunt comparaţi pentru a determina unde se uită utilizatorul:

- Imaginea ochiului stâng este pusă in oglindă (b) şi este scăzută din imaginea ochiului drept (a) $\Rightarrow$ (c & d).

- Dacă utilizatorul se va uita direct la cameră diferenţa este mică (d). Daca utilizatorul se uită în partea stângă, atunci ochiul pus în oglindă va părea ca se uită în partea dreaptă (b) iar diferenta (c) este evidenta.

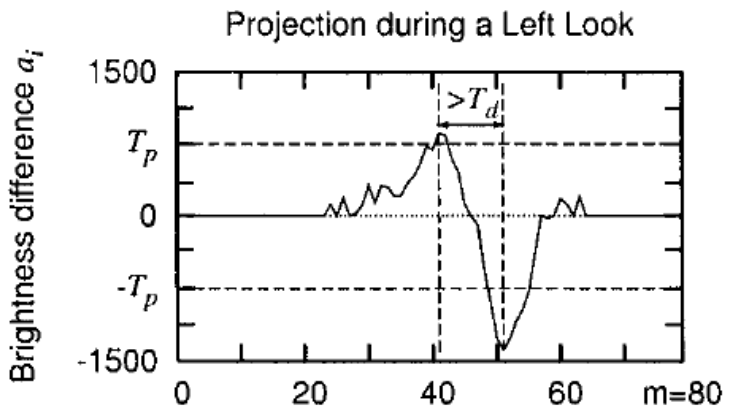
-Diferentele de intensitate (cu semn) dintre imaginea ochiului drept si cel stang oglindit sunt proiectate (insumate) pe verticala $\Rightarrow$ masurarea directiei privirii

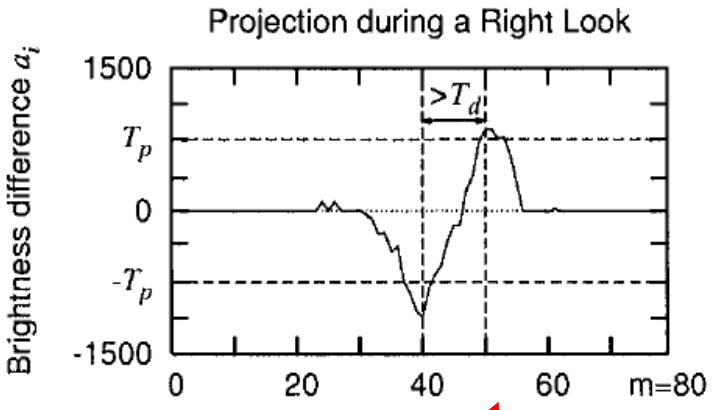


a) Ochiul drept (direcţia de privire = stânga).
b) Imagine în oglindă a ochiului stâng.
c) Img. diferenta (fara semn): la privire îndreptată în stânga.
d) Img. Diferenta (fara semn): la privire îndreptată înainte.

Analiza directiei privirii

4. Masurarea directiei privirii





$$a_i = \sum_{j=1}^{n} (I_r(i,j) - I_\ell(m-i,j))$$

$$a_{\min} = \min_{i=\{1,\ldots,m\}} (a_i) \qquad i_{\min} = \arg\min_{i=\{1,\ldots,m\}} (a_i)$$

$$a_{\max} = \max_{i=\{1,\ldots,m\}} (a_i) \qquad i_{\max} = \arg\max_{i=\{1,\ldots,m\}} (a_i)$$

Exista miscare a ochilor: $a_{\max} > T_p$ $a_{\min} < -T_p$

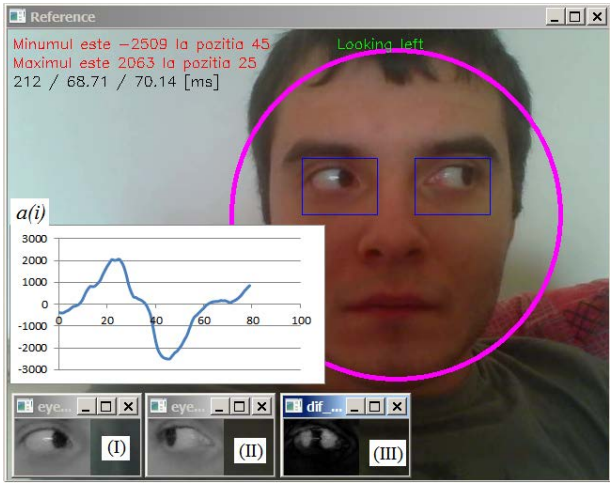
Directia miscarii:

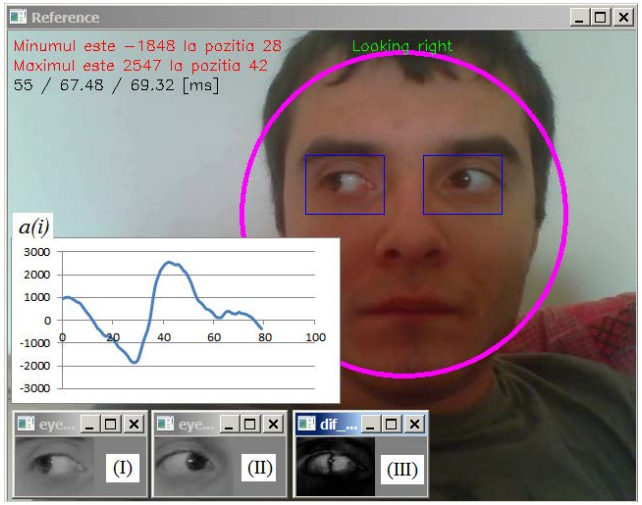
$i_{\max} - i_{\min} > T_d \Rightarrow$ 'right motion'

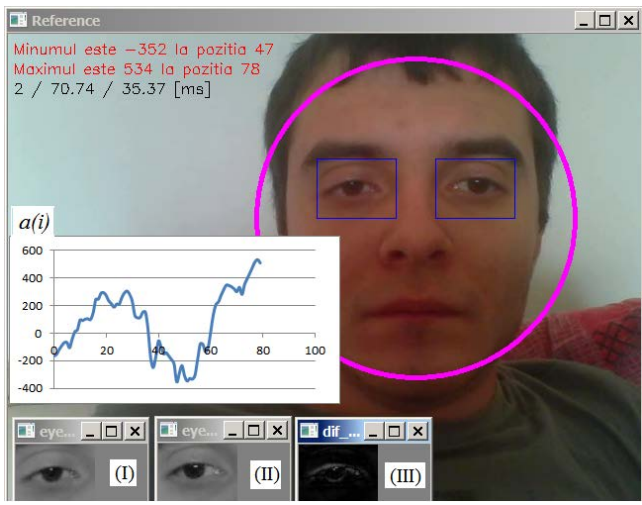
$i_{\max} - i_{\min} < -T_d \Rightarrow$ 'left motion'

Analiza directiei privirii

Exemple:







a. b. c.

Ilustrarea rezultatelor detectiei privirii: (a) spre stanga, (b) spre dreapta (c) neutra/fata.

Detectia de compnente faciale

Detecti si urmarirea ochilor.

Detectia clipitului.

Referinte

M. Chauand, M. Betke, Real Time Eye Tracking and Blink Detection with USB Cameras, Boston University, Computer Science, Technical Report No. 2005-12.

K.Grauman, M.Betke, J.Gips, G.Bradski, Communication via eye blinks detection and duration analysis in real time. Proceedings of the IEEE Computer Vision and Pattern Recognition Conference (CVPR2001), Vol.2,pages1010–1017,Kauai, Hawaii, December 2001.

Aplicatii

Aplicabilitate

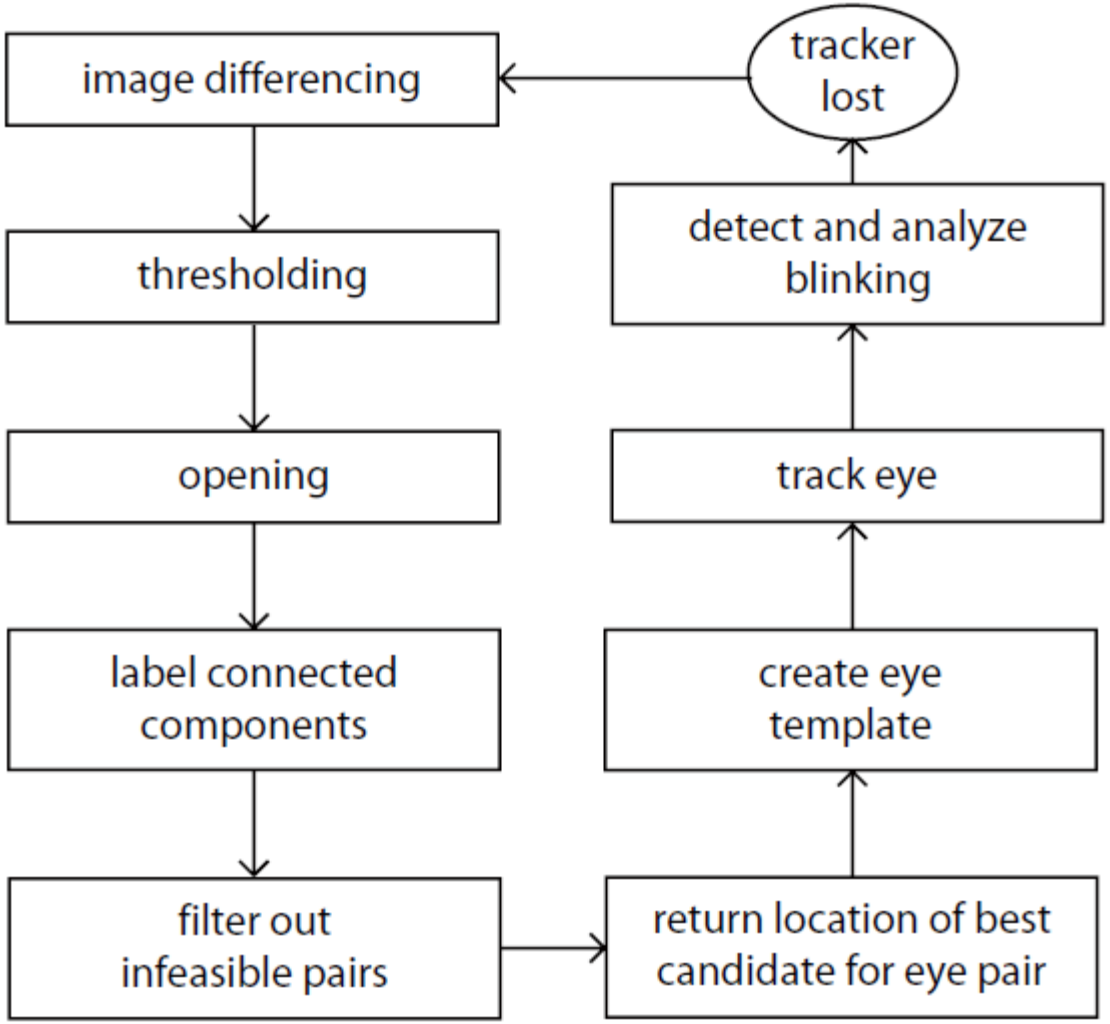
Interfete pentru persoane cu dizabilitati
- detectia sablonului de clipire $\Rightarrow$ interpretare actiuni

Sistem de detectie a gradului de atentie
- Aplicatii de asistenta a conducerii

Liveness detection
- Siteme de securitate biometrice (bazate pe detectia si recunoasterea fetelor)

Performante
- Hardware minimal (WebCam)
- Perfomante de timp real:
  30 fps, 320x240, P4 2.8GHz
  implementare cu functii OpenCV

Scama bloc a aplicatiei



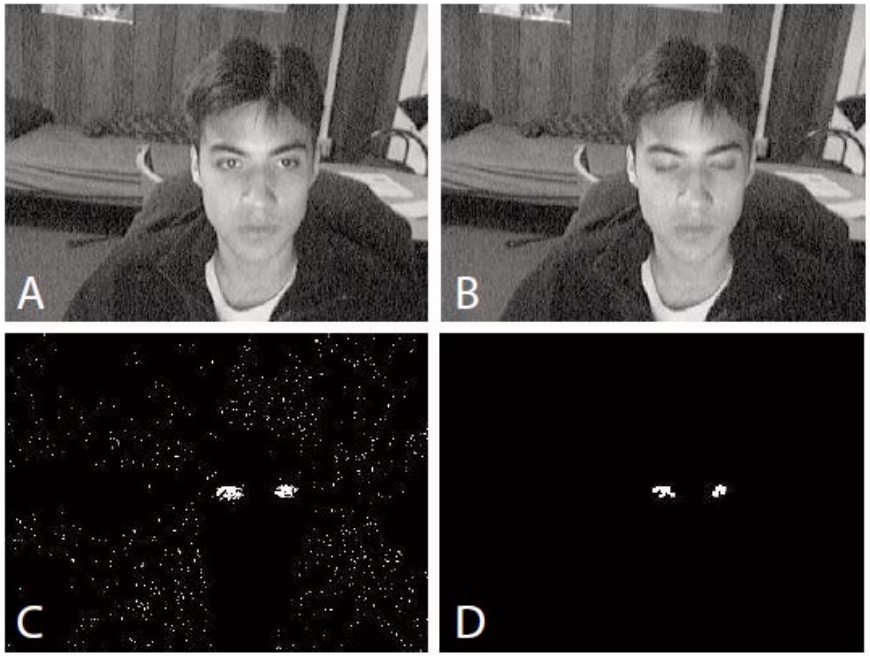
Initializare

1. Imaginea diferenta (B-A) ⇒ binarizare: C

```
cvSub(gray, prev, diff, NULL);
cvThreshold(diff, diff, 5, 255, CV_THRESH_BINARY);
```

2. Eliminare zgomote ⇒ deschidere: D

```
IplConvKernel* kernel;
kernel = cvCreateStructuringElementEx(3, 3, 1, 1, CV_SHAPE_CROSS, NULL);
cvMorphologyEx(diff, diff, NULL, kernel, CV_MOP_OPEN, 1);
```



Initializare

3. Etichetare $\Rightarrow$ detectie componente conexe

- dc. nr. componente conexe este prea mare (miscare pronuntata a capului) $\Rightarrow$ reinitializare

4. Filtrare suplimentara a componentelor conexe $\Rightarrow$ perechi cu urmatoarele proprietati (ochii):

- latime/inaltime asemenatoare
- deplasament pe verticala mic
- deplasamanet pe orizontala proportional cu dimensiunea componentelor conexe (trasaturi antropomorfice

5. Determinare coordonate dreptunghi circumscris la cea mai mare eticheta din pereche (ochiul cel mai bine vizibil) $\Rightarrow$ ROI

Generare template

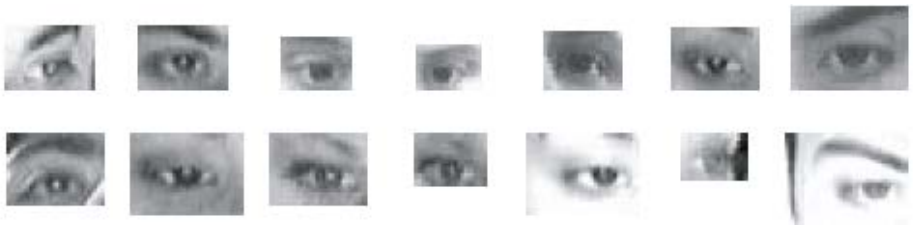
Template-ul pt. ochi se creeaza on-line in faza de initializare

Template-ul se copiaza din regiunea de interes (ROI) selectata la pasul precedent la momentul $t+\Delta t$ (t momentul in care se calculeaza segmentarea bazata pe clipire)

Intarzirea Δt este mecesara pt. stabilizarea imaginii ochiului (ochi deschis). Valoarea ei trebuie sa fie mai mica decat intervalul de timp dintre 2 clipiri consecutive involuntare pt. a achizitiona imaginea ochiului deschis

Template-ul ochiului deschis se va folosi in tracking-ul ochilor.

```
cvWaitKey(250);
cvSetImageROI(gray, rect_eye);
cvCopy(gray, tpl, NULL);
cvResetImageROI(gray);
```



Urmarirea ochilor

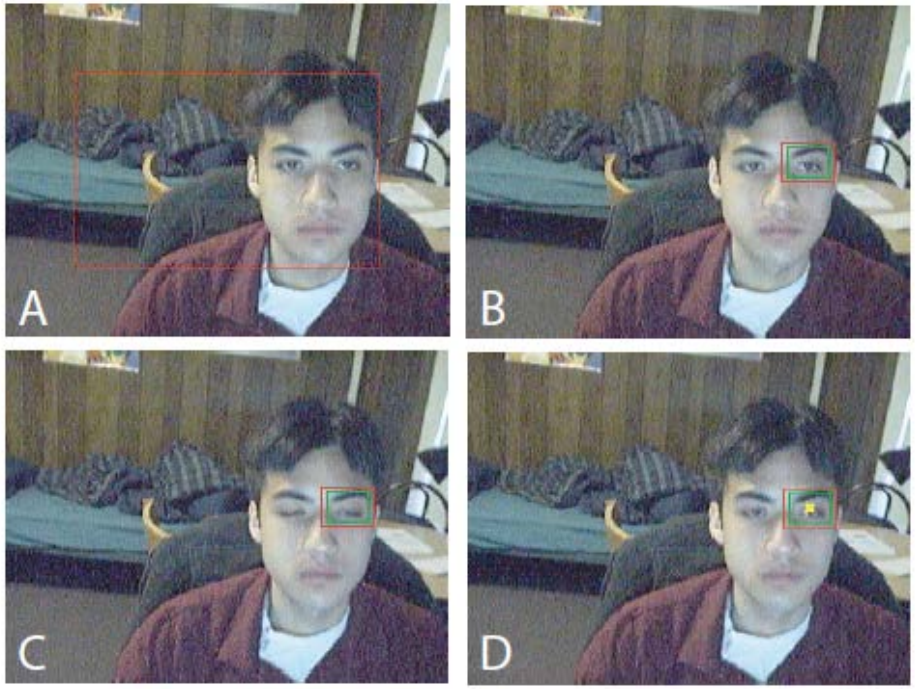
Rafinarea pozitiei ochilor in fiecare frame pe baza de template-matching

- nu se foloseste nici o metoda de filtrare (ex. Kalman) !

Metrica de corelatie: corelatia normalizata (robusta la variatii de iluminare)

$$\frac{\sum_{x,y}[f(x,y)-\bar{f}_{u,v}][t(x-u,y-v)-\bar{t}]}{\sqrt{\sum_{x,y}[f(x,y)-\bar{f}_{u,v}]^2 \sum_{x,y}[t(x-u,y-v)-\bar{t}]^2}}$$

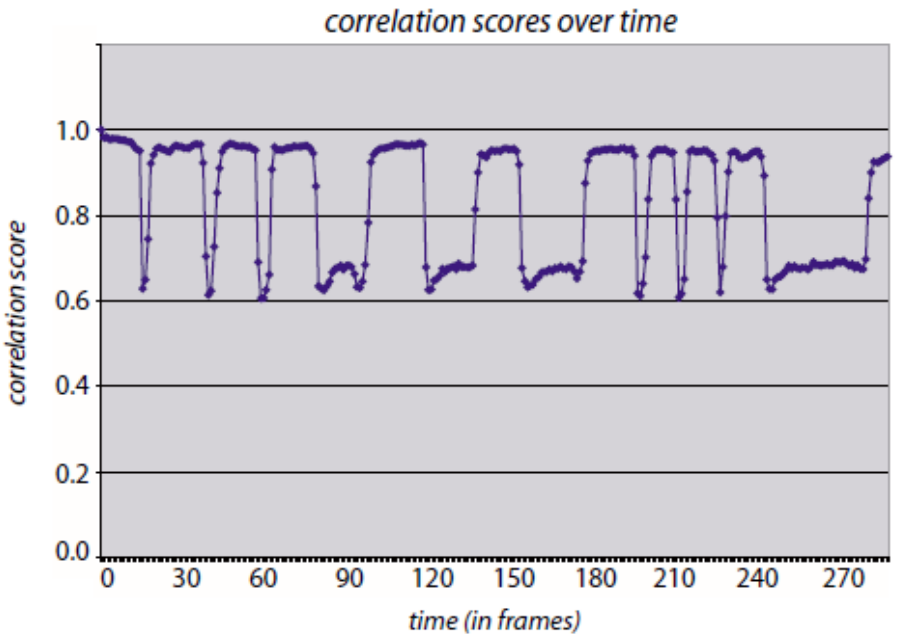
Potrivire $\rightarrow 1$




```
/* get the centroid of eye */
point = cvPoint(
    rect_eye.x + rect_eye.width / 2,
    rect_eye.y + rect_eye.height / 2
);
/* setup search window */
window = cvRect(
    point.x - WIN_WIDTH / 2,
    point.y - WIN_HEIGHT / 2,
    WIN_WIDTH,
    WIN_HEIGHT
);
/* locate the eye with template matching */
cvSetImageROI(gray, window);
cvMatchTemplate(gray, tpl, res,
CV_TM_SQDIFF_NORMED);
cvMinMaxLoc(res, &minval, &maxval, &minloc,
&maxloc, 0);
cvResetImageROI(gray);
```

Detectia clipirii

Calculeaza scorul de corelatie in fiecare frame



Masurarea frecventei de clipire:

-masurarea timpului T_D (ochi deschis si T_I (ochi inchis)

-masurarea se face prin declanasarea de timere cand functia de corelatie coboara/creste sub/peste pragurile stabilite

Pragurile $\Rightarrow$ statistici pt. mai multi utilizatori (valabile doar daca capul ramane nemiscat)

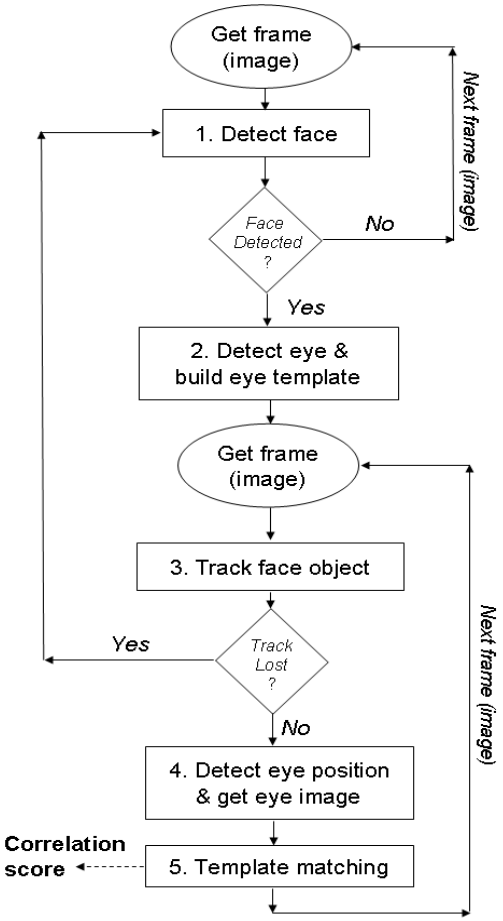
- 0.85 ... 1: ochi deschis

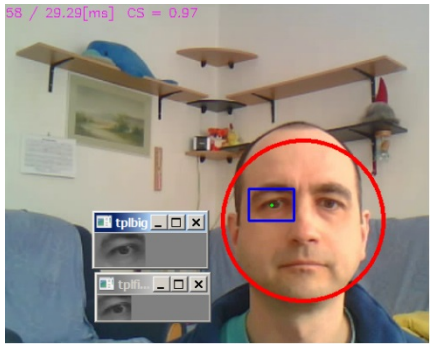
- 0.6 ... 0.7 ochi inchis

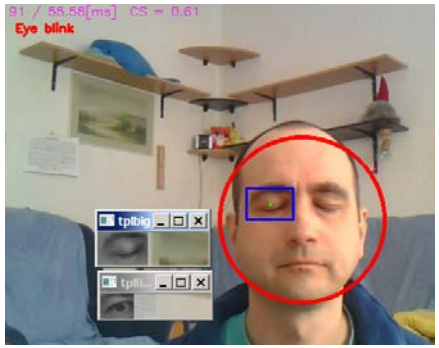
- sub 0.45: tracking pierdut $\Rightarrow$ reinitializare tracking

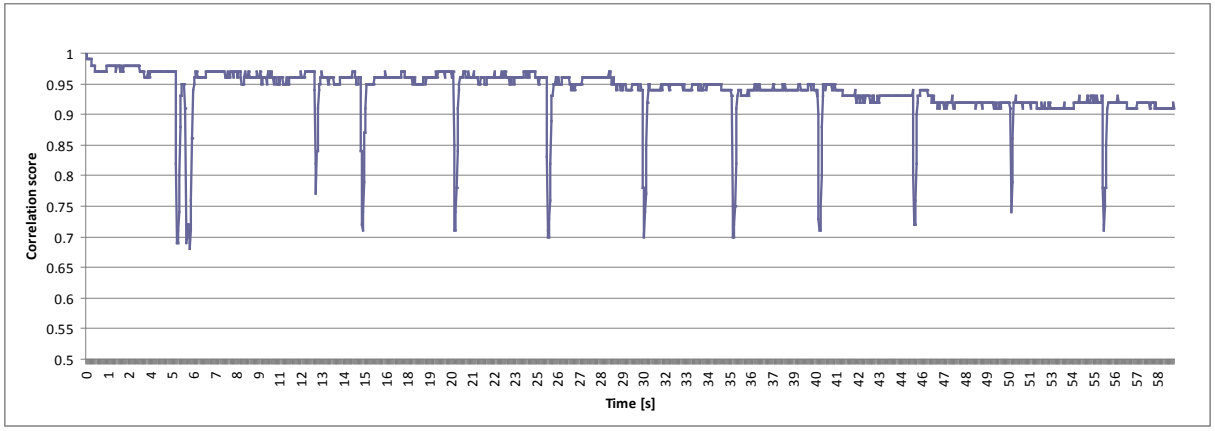
Detectia clipirii

Varianta a metodei folosind detectia fetei (Viola-Jones) si a pozitiei ochilor pe baza trasaturilor antropomorfice:









Ilustrare a functiei de corelatie pentru un sablon de clipire normal (≈12 clipiri/min).

Postprocesare detectie fete

Validare detectie

Se pate face pe baza urmatoarelor trasaturi

- culoare
- trasaturi antropomorfice (pozitie relativa trasaturi faciale)
- factor de simetrie

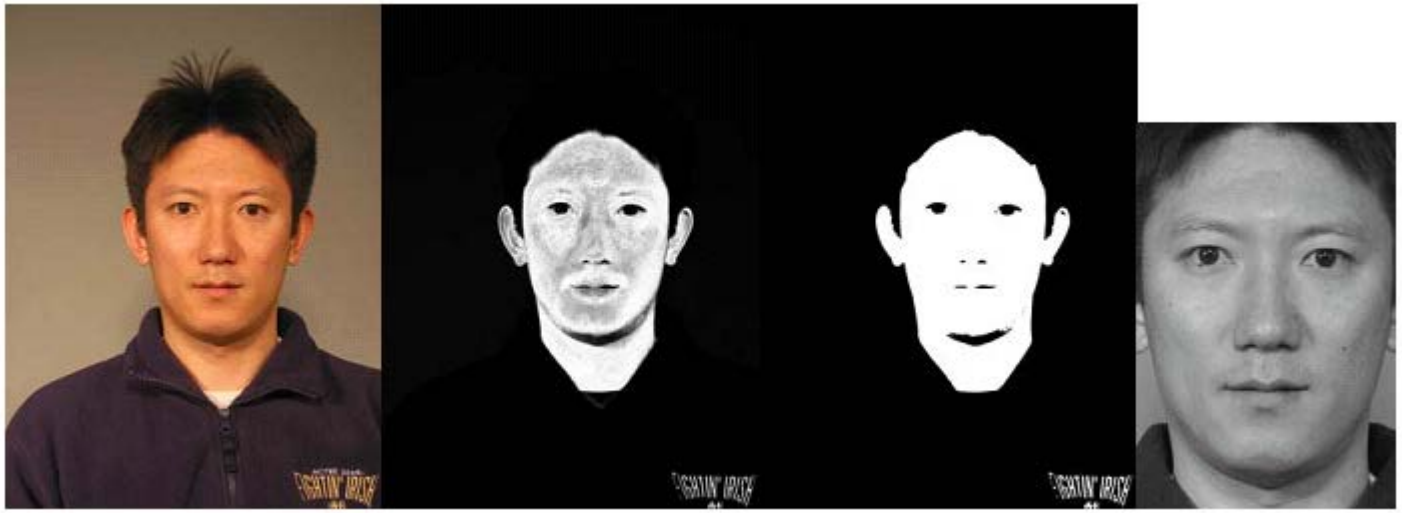
Detectie orientare

- Rotatie in planul imaginii

Detectia axei de simetrie si orientarii

X. Chen, P.J. Flynn, K.W. Bowyer, “Fully Automated Facial Symmetry Axis Detection in Frontal Color Images”, Department of Computer Science and Engineering University of Notre Dame, Notre Dame, IN 46556 USA,

Metoda de detectie a fetelor – similara cu metoda 1



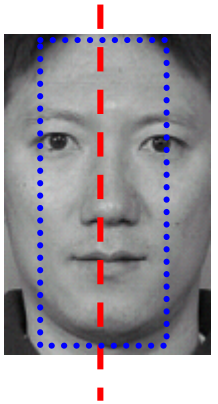
Detectia axei de simetrie

1. Se considera o axa se simetrie initiala (ex. axa de alungire a fetei segmentate sau axa vericala ce trece prin centrul de masa)

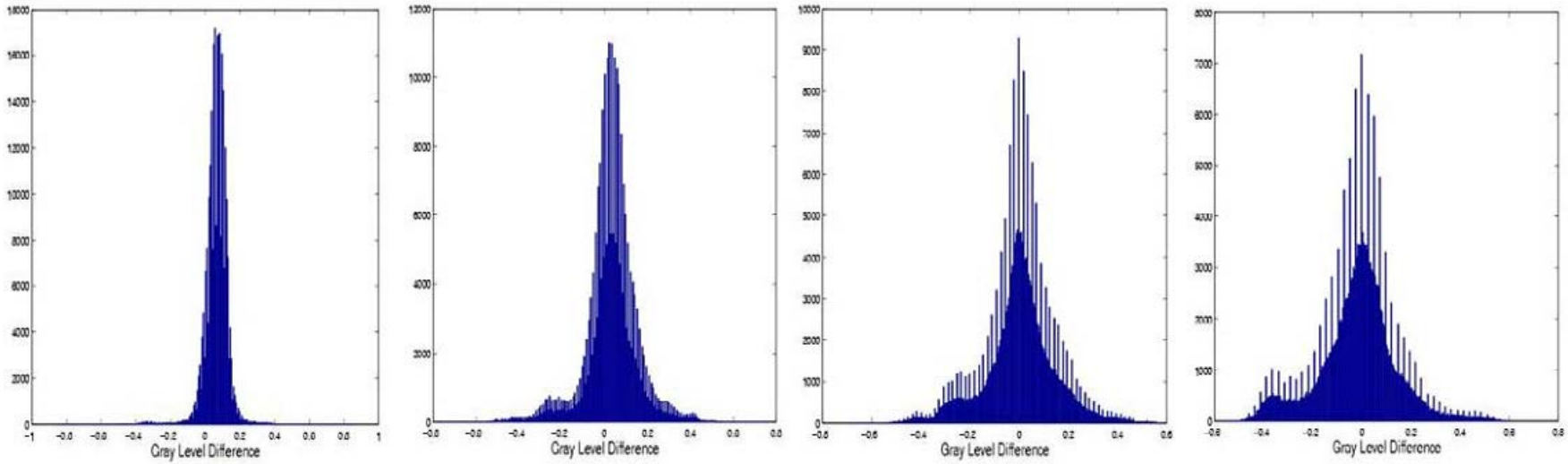
2 Se calculeaza gray level difference histogram (GLDH) in jurul axei considerate:

   - calculeaza dferentele de intnsitate intre perechi de pixeli simetrici in jurul axei $\Rightarrow$ histograma (GLDH)

3. Se ajusteaza discret pozitia axei de simetrie: in pozitia corecta media (varful histogramei) este maxima si deviata standard este minima



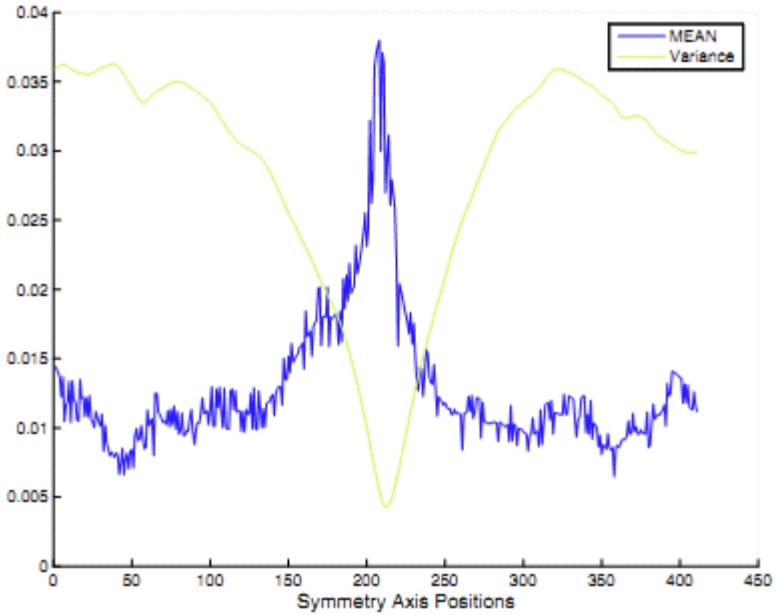
Deviatie: 0 20 40 60



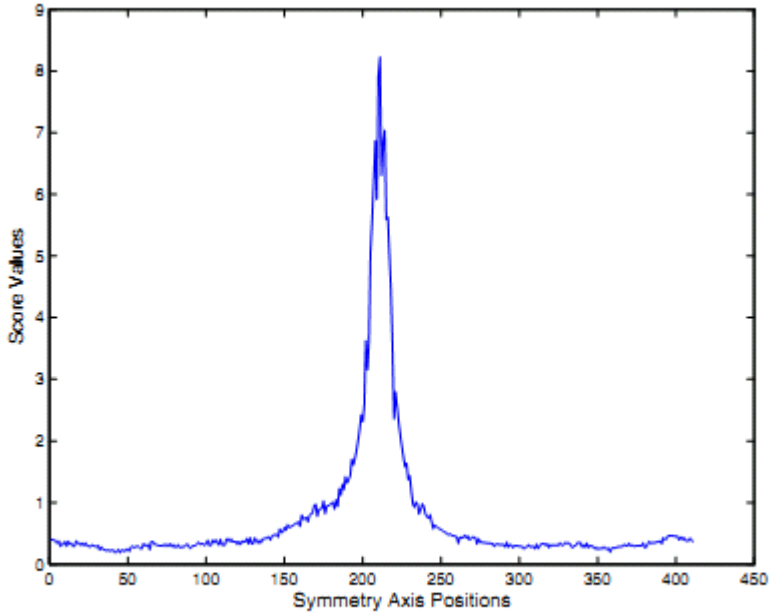
Detectia axei de simetrie

Pt. gasirea pozitiei optime (media (varful histogramei) este maxima si deviata standard este minima) se calculeaza *Y-score* si se maximizeaza in functie de pozitie:

$$Y = \frac{MEAN}{Variance}.$$



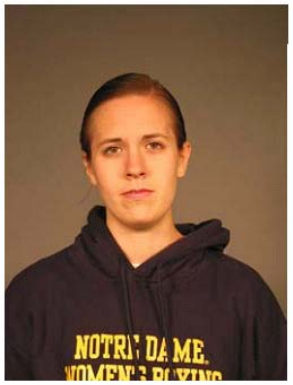
(a) MEAN and variance

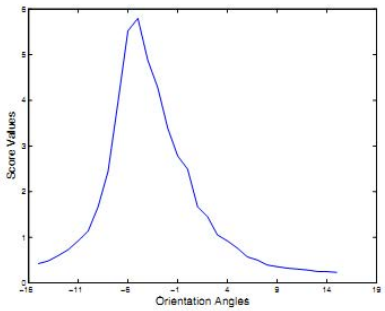


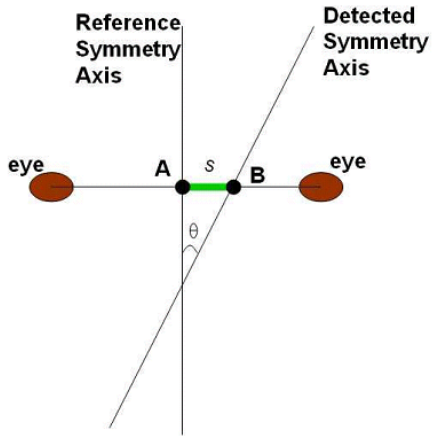
(b) Y score

Detectia orientarii

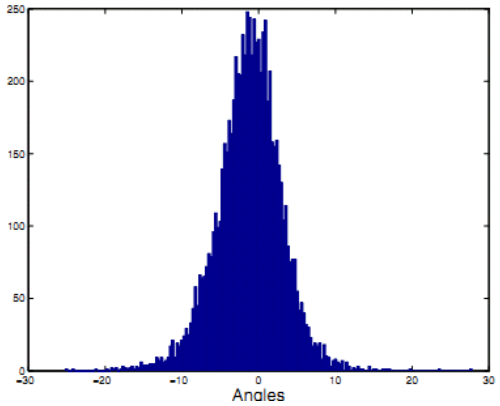
Se calculeaza *Y-score* pt. orientari in intervalul -15 … 15 grade cu pas de 1 grad.



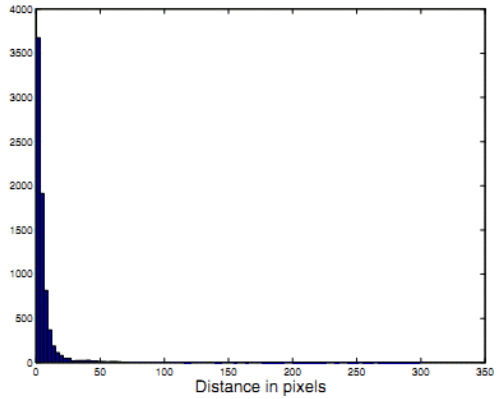




Rezultate (7500 imagini) – comparatie cu GT selectat manual



Eroare unghi



Eroare pozitie