Cuprins

[PREZENTARE GENERALĂ 2](#_Toc187712004)

[1. 1 Histograma gradienților orientați (HOG) 2](#_Toc187712005)

[1. 2 Trăsătura HAAR 5](#_Toc187712006)

[1. 3 Detectarea Feței și Landmark-urilor 7](#_Toc187712007)

[1. 4 Bibliografie 7](#_Toc187712008)

[IMPLEMENTAREA DETECȚIEI STĂRII EMOȚIONALE BAZATE PE TRĂSĂTURILE FACIALE 8](#_Toc187712009)

[2. 1 Materiale și Metode 8](#_Toc187712010)

[2. 2 Implementarea algoritmului (parte practică) 10](#_Toc187712011)

[2. 3 Rezultate 13](#_Toc187712012)

[2. 4 Bibliografie 14](#_Toc187712013)

|  |  |
| --- | --- |
|  | PREZENTARE GENERALĂ |

### 1. 1 Histograma gradienților orientați (HOG)

Algoritmul HOG (Histogram of Oriented Gradients) este o tehnică folosită pentru extracția de caracteristici din imagini, utilizată frecvent în recunoașterea de obiecte și clasificarea imaginilor.

Aceasta tehnică numară de câte ori apare o orientare anume a gradientului într-o anumită regiune din imagine.

HOG analizează structura locală a unei imagini, concentrându-se pe contururi și margini. Ideea de bază este că forma și structura unui obiect pot fi reprezentate prin distribuția gradientelor de intensitate și orientările lor.

Trăsăturile HOG au fost introduse de Navneed Dalal si Bill Triggs [1] care au dezvoltat și au testat mai multe variante de descriptori HOG, cu diferite organizări spațiale, diferite metode de calcul a gradienților, și diferite metode de normalizare.

Ideea esențială care stă la baza descriptorilor HOG este ca aspectul local si forma unui obiect într-o imagine poate fi descrisă de distribuția intensității gradienților și de distribuția orientării muchiilor. Implementarea acestor descriptori poate fi obținută prin împărțirea imaginii în regiuni conexe mai mici, numite celule, și calcularea, pentru fiecare celulă, a histogramei orientării muchiilor (sau a direcției gradienților) pe baza pixelilor aparținând celulei. Combinarea histogramelor din celule reprezintă descriptorul imaginii. [1]

Pentru o mai buna performanță, histogramele locale pot fi normalizate prin calcularea unei măsuri a intensității pe o regiune mai mare a imaginii, numită bloc, și apoi folosind această măsură pentru normalizarea tuturor celulelor din acel bloc. Această normalizare are ca efect o invarianță sporită la schimbarea iluminarii, sau la umbre.

Primul pas pentru obținerea HOG este calculul valorilor gradientului. Cea mai folosită metodă este aplicarea maștilor de derivare unidimensională pe direcție verticală și pe direție orizontală, adică filtrarea imaginii cu următoarele două nuclee de convoluție:

(1.1)

O imagine care conține roată, vehicul, Vehicul de teren, anvelopă

Descriere generată automat

Fig. .. Imagine inițială

**O imagine care conține schiță, desen, vehicul, Vehicul de teren

Descriere generată automat**

**O imagine care conține schiță, vehicul, desen, Vehicul de teren

Descriere generată automat**

Fig. .. Derivata pe orizontală

Fig. .. Derivata pe verticală

Dându-se o imagine *I*, vom obține derivată orizontală si derivată verticală folosind un operator de convoluție:

(1.2)

Magnitudinea gradientului este:

(1.3)

Orientarea gradientului este:

(1.4)

Al doilea pas constă în crearea histogramelor pentru fiecare celulă. Fiecare pixel din fiecare celulă are un vot în histogramă. Poziția în histogramă care va fi incrementată de un anume pixel este dată de valoarea orientării gradientului acestui pixel. Dalal și Triggs au găsit că este suficient ca histograma să aibă 9 poziții, iar dacă orientarea gradientului ia valori între 0 și 360 grade, poziția în histogramă este calculată prin împarțirea cu 40. În procesul de votare fiecare vot poate fi ponderat cu valoarea magnitudinii gradientului, cu rădăcina pătrată sau cu pătratul magnitudinii. [1]

O imagine care conține roată, vehicul, Vehicul de teren, mașină

Descriere generată automat

Fig. .. Magnitudinea gradientului

**O imagine care conține roată, mașină, vehicul, Vehicul de teren

Descriere generată automat**

Fig. .. Împărțirea în celule

A group of colorful lines

Description automatically generated

Fig. .. Histograma orientarii gradientului

Pentru a ține cont de variațiile de iluminare și contrast, magnitudinea gradienților trebuie normalizată local, ceea ce înseamnă gruparea celulelor în structuri spațiale mai mari, numite blocuri. Descriptorul HOG este vectorul format din componentele normalizate ale histogramelor din fiecare celulă din bloc. De obicei blocurile se suprapun parțial, astfel ca fiecare celulă contribuie de mai multe ori la descriptorul final. Exista două tipuri principale de blocuri: blocul rectangular, R-HOG, și blocul circular, C-HOG. Blocurile R-HOG sunt grile pătrate, definite de trei parametri: numărul de celule din bloc, numărul de pixeli într-o celula, și numărul de poziții în histograma unei celule.

### . 2 Trăsătura HAAR

Trăsăturile Haar sunt modele simple care compară intensitatea pixelilor din anumite regiuni ale unei imagini. Acestea detectează modele sau schimbări în intensitatea pixelilor, care corespund contururilor sau structurilor geometrice caracteristice obiectelor. Provin din transformata Wavelet de tip Haar. O trăsătură Haar este o combinație de regiuni **albe** și **negre** aranjate într-un dreptunghi. [2]

Valoarea unei trăsături Haar este calculată ca diferența dintre suma intensităților pixelilor din zona albă și suma intensităților pixelilor din zona neagră.

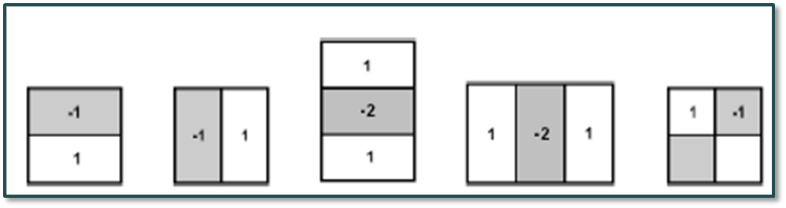


Fig. .. Transformata Wavelet de tip Haar

Lienhart propune 45 trăsături dreptunghiulare rotate. [2]

O trăsătură este reprezentată de:

* Tipul trăsăturii: 1(a), 2(b);
* Locația (x, y)
* Dimensiunea trăsăturii (w, h)

Pentru o fereastră 24x24 există 117,941 trăsături diferite

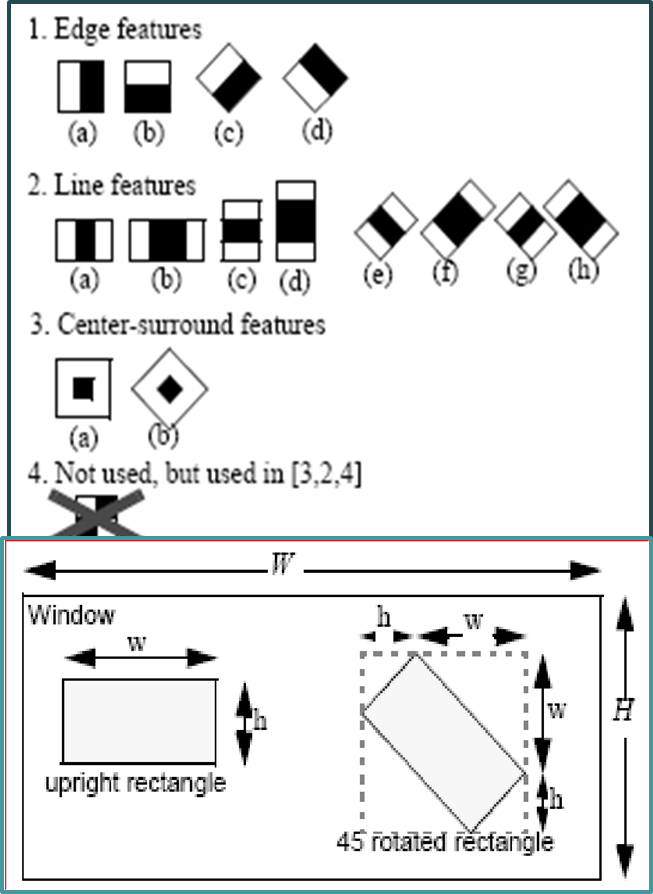


Fig. .. Trăsături Haar extinse

### . 3 Detectarea Feței și Landmark-urilor

Facial landmarks sunt puncte cheie care definesc trăsăturile esențiale ale feței (sprâncene, ochi, nas, gură, contur facial).

Se utilizează un model preantrenat “shape\_predictor\_68\_face\_landmarks.dat” pentru predicția a 68 de puncte cheie, bazat pe lucrarea lui Kazemi și Sullivan (One Millisecond Face Alignment with an Ensemble of Regression Trees). [3]

Distribuția punctelor se realizeaza astfel.

* 17 puncte pentru conturul feței.
* 10 puncte pentru sprâncene.
* 12 puncte pentru ochi.
* 9 puncte pentru nas.
* 20 puncte pentru gură (inclusiv buzele).

Landmark-urile oferă coordonate precise care permit calcularea distanțelor și raporturilor geometrice utilizate pentru analiza emoțională. Schimbările pozițiilor relative ale landmark-urilor reflectă expresiile emoționale (ex. colțurile gurii pentru fericire, sprâncenele pentru furie).

### . 4 Bibliografie

|  |  |
| --- | --- |
| [1] | N. T. B. Dalai, „Histograms of oriented gradients for human detection,” *Computer Society Conference on Computer Vision and Pattern Recognition,* 2005. |
| [2] | C. Florea, „Metode de recunoaștere generică,” *Image Processing and Analysis Laboratory,* 2018. |
| [3] | H. G. a. A. C. E. Sariyanidi, „Automatic Analysis of Facial Affect: A Survey of Registration, Representation, and Recognition,” *Transactions on Pattern Analysis and Machine Intelligence,* pp. 1113-1133, 2015. |

|  |  |
| --- | --- |
|  | IMPLEMENTAREA DETECȚIEI STĂRII EMOȚIONALE BAZATE PE TRĂSĂTURILE FACIALE |

### . 1 Materiale și Metode

Oamenii sunt destul de buni la detectarea emoțiilor din elementele faciale (acestea sunt sprâncene, ochi, nas și gură). Reperele pot fi detectate. Potrivit , detectarea lor este un „subset al problemei predicției formei”.

Primul pas în detectarea reperelor faciale este detectarea feței. Apoi se construiește o regiune de interes (ROI). Al doilea pas este detectarea structurii faciale cheie. Detectorul este implementat în biblioteca „dlib”.[1][2]

O imagine care conține Grafică

Descriere generată automat

Fig. .. Elementele feței detectate și numerotate (fiecare marcaj are un număr)

În Figura 2.1, se pot vedea reperele faciale care descriu sprâncenele, ochii, nasul, gura și întreaga față. Fiecare reper facial în total) are un număr și o poziție în imagine.

O imagine care conține schiță, desen, Schiță, clipart

Descriere generată automat

Fig. .. O faţă umană neutră

O față neutră (Figura 2.2) poate fi descrisă astfel: trăsături echilibrate, fără expresii evidente. Sprâncenele sunt relaxate, ochii sunt deschiși dar fără o privire intensă, iar buzele sunt ușor închise sau într-o poziție naturală, fără zâmbet sau încruntare. Pielea este uniformă, iar mușchii feței nu exprimă nicio emoție clară.

O față fericită (Figura 2.3) are trăsături luminoase, sprâncenele sunt ușor ridicate sau relaxate, iar ochii par mai strălucitori, adesea cu riduri de expresie la colțuri. Buzele sunt întinse într-un zâmbet larg sau subtil, iar obrajii pot fi ușor ridicați. Întreaga expresie transmite o stare de bucurie și căldură.

. O imagine care conține schiță, desen, Schiță, clipart

Descriere generată automat

Fig. .. O faţă umană fericită

O față furioasă (Figura 2.4) are trăsături tensionate: sprâncenele sunt adunate și coborâte spre centru, formând o încruntare pronunțată. Ochii sunt îngustați și privirea este intensă sau fixă. Nasul poate fi ușor încrețit, iar buzele sunt strânse, uneori cu colțurile guri trase în jos sau dinții încleștați. Mușchii feței sunt încordați, iar expresia generală transmite iritare sau agresivitate.

O imagine care conține schiță, desen, Schiță, ilustrație

Descriere generată automat

Fig. .. Faţă umană furioasă

Pe baza observației de mai sus, se poate observa importanța elementului facial în detectarea stării emoționale. Metoda propusă este de a determina cazurile particulare care reflectă stările emoționale de mai sus folosind reperele faciale prezentate în figura 2.1. [3]

### . 2 Implementarea algoritmului (parte practică)

from cv2 import FONT\_HERSHEY\_SIMPLEX, imshow, waitKey, putText, destroyAllWindows, imread, cvtColor, COLOR\_BGR2GRAY, \  
 imwrite, circle  
# Importare biblioeci  
import dlib  
import cv2  
  
def print\_hi(name):  
 print(f'Hi, {name}') # Press Ctrl+F8 to toggle the breakpoint.  
  
def main():  
 global landmarks\_list  
 detector = dlib.get\_frontal\_face\_detector()  
 predictor = dlib.shape\_predictor("shape\_predictor\_68\_face\_landmarks.dat")  
  
 # prag  
 thold = 0.18  
 angThold = 0.07  
  
#Imaginile utilizate pentru recunoastere  
  
 #filename = 'einstein'  
 filename = 'happy\_man01'  
 #filename = 'woman00'  
 #filename = 'woman02'  
 #filename = 'woman06'  
 #filename = 'woman08'  
 #filename = 'trudeau01'  
 #filename = 'angryDonald01'  
 #filename = 'angryDonald'  
 #filename = 'angry\_man02'  
 #filename = 'angry\_man01'  
  
 # incarca imaginea originala  
 img = imread('./FACES/' + filename + '.jpg', 1)  
  
 # Afiseaza imaginea originala  
 imshow('Original', img)  
  
 scale\_percent = 55 # Scalează la 50% din dimensiunea originală  
 width = int(img.shape[1] \* scale\_percent / 100)  
 height = int(img.shape[0] \* scale\_percent / 100)  
 dim = (width, height)  
 img = cv2.resize(img, dim, interpolation=cv2.INTER\_AREA)  
  
 # converteste in alb-negru  
 img\_gray = cvtColor(img, COLOR\_BGR2GRAY)  
  
 # detecteaza fata  
 faces\_in\_image = detector(img\_gray, 0)  
  
 # bucla  
 for face in faces\_in\_image:  
  
 # atribuie facial landmarks  
 landmarks = predictor(img\_gray, face)  
  
 # pune coordonatele celor 68 de puncte in lista  
 landmarks\_list = []  
 for i in range(0, landmarks.num\_parts):  
 landmarks\_list.append((landmarks.part(i).x, landmarks.part(i).y))  
  
 # noteaza fiecare landmark cu un numar  
 for landmark\_num, xy in enumerate(landmarks\_list, start=1):  
 circle(img, (xy[0], xy[1]), 10, (0, 0, 0), -1)  
 # cv2.circle(img, (xy[0], xy[1]), 12, (168, 0, 20), -1)  
 putText(img, str(landmark\_num), (xy[0] - 7, xy[1] + 5), FONT\_HERSHEY\_SIMPLEX, 0.4, (255, 255,), 1)  
 # cv2.imwrite('./FACES/woman00\_proc.jpg', img)  
  
 # stabileste caracteristicile textului  
 textPlace = (img.shape[0] / 2, 0)  
 font = FONT\_HERSHEY\_SIMPLEX  
 fontScale = 1  
 fontColor = (255, 0, 0)  
 lineType = 1  
  
 # Emotion  
 L1VertREL = (landmarks\_list[58][1] - landmarks\_list[52][1]) / (landmarks\_list[9][1] - landmarks\_list[28][1])  
 EyeVertREL = (landmarks\_list[42][1] - landmarks\_list[38][1]) / (landmarks\_list[9][1] - landmarks\_list[28][1])  
 if (L1VertREL > thold):  
 if (EyeVertREL < angThold):  
 text = "Angry Face"  
 # print("Happy Face")  
 else:  
 text = "Happy Face"  
  
  
 if (L1VertREL < thold):  
 text = "Neutral Face"  
 # print("Neutral Face")  
  
 putText(img, text,  
 (10, 50),  
 FONT\_HERSHEY\_SIMPLEX,  
 1,  
 255)  
  
 # salveaza imaginea cu landmarks  
 imwrite('./Faces/' + filename + '\_proc.jpg', img)  
 print(img.shape)  
  
 # afiseaza imaginea cu landmarks  
 imshow('Procesata', img)  
  
 waitKey(0)  
 destroyAllWindows()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 #print\_hi('PyCharm')  
 main()

### . 3 Rezultate

După cum se poate observa, în imaginile de mai jos s-a demonstrat funcționarea algorimtului de recunoaștere a expresiilor faciale implementat.

O imagine care conține Chip de om, portret, rid, captură de ecran

Descriere generată automat

Fig. .. Imaginea originală

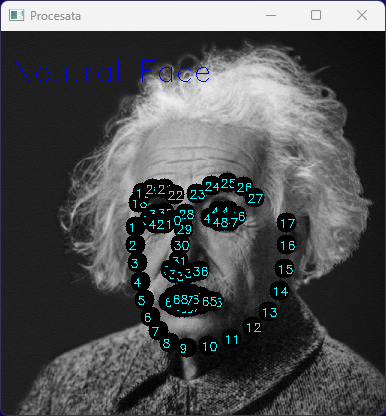


Fig. .. Imagine procesată

### 2. 4 Bibliografie

|  |  |
| --- | --- |
| [1] | R. A., „Facial landmarks with dlib, OpenCV, and Python,” 2017. |
| [2] | A. R. S. I. A. D. K. K. a. S. K. T. Balomenos, „Emotion Analysis in Man-Machine Interaction Systems,” *Procceedings of the First International Workshop on Machine Learning for Multimodal Interaction,* p. 318–328, 2005. |
| [3] | S. J. Kazemi V., „One millisecond face alignment with an ensemble of regression trees,” *Conference on Computer Vision and Pattern Recognition,* 2014. |