## Proiect la procesarea imaginilor

Tematica proiectului:

Recunoașterea de flăcări în materiale video (Flame Recognition). Detecția vizuală a focului

poate fi aplicată în condiții în care metodele convenționale de detectare nu se pot aplica,

cum ar fi recunoașterea de flăcări în materiale vide filmat. Se cere detectarea flăcărilor în

secvențe video folosind informații de culoare și mișcare. Se pot folosi segmentare bazată pe

culoare, eroziune, region growing, optical flow.

## Introducere

Recunoasterea flacarilor in materiale video este o tehnica care devine tot mai eficienta comparativ cu metodele conventionale de detectare a flacarilo. Motivul pentru aceasta schimbare este faptul ca alarmele conventionale nu pot fi folosite in toate situatiile si au posibilitatea sa se declanseze in situatii nedorite.

Ca urmare am dezvoltat o aplicatie in python pentru a detecta flacarile din materialul video. Aplicatia se bazeaza pe color detection si edge detection pentru a calcula posibilitatea unui foc bazata pe fluctuatia volumului acestuia. Pentru a elimina cat mai mult alarmele nedorite am utilizat diferite metode de procesare a imaginii video precum si conditii logice.

Python a fost utilizat in colaborare cu openCV pentru toate metodele de procesare a imaginii video si numpy pentru operatii ce includ matrici.

Programul salveaza la finalul unei executii 2 fisiere video, unul cu filtrarea initiala a flacarii din imagine si unul cu conturul de detectie pozitionat asupra flacarilor din clipul original, precum si un fisier text care contine sansa de foc la fiecare moment al clipului si sansa maxima obtinuta. Pe baza acestei sanse de foc putem porni alarma de incendiu.

## Executie

Programul trece prin urmatorii pasi aplicandui pe fiecare cadru din video:

1. Filtreaza culoarea focului din imagine folosind 2 valori alese una din care este calibrata usor la inceputul videoului pentru micsorarea de artefacte
2. In functie de cantitatea de culoare obtinuta trecem in etapa de procesare in care erodam, limitam si dilatam imaginea pentru a obine zona generala a flacarii independenta de intensitatea sa
3. Folosim contour detection pentru a face rost de volumul vizual al flacarii
4. Filtram contururile de dimensiuni foarte mici care au fost receptionate
5. Calculam fluctuatiile flacarii bazandune pe cadrul antecedent. Astfel putem vedea masurand perimetrul conturului daca flacara a crescut in marime, daca are aceeasi marime sau daca marimea ei este stabila
6. Pe baza acestor date consideram daca cadrul respectiv are o flacara in el sau nu.
7. Apoi, cu aceasta ultima informatie constituim nivelul de incredere in existenta unei flacari folosind un istoric de 50 de cadre care la 30fps are o durata de aprox. 1.66 secunde

Slabiciunea acestui sistem poate fi un foc controlat exceptional dar si posibilitatea de a se activa pe baza unui obiect strain care are nuanta de portocaliu caracteristica focului si care indeplineste conditiile de miscare alese pentru a filtra informatia. De aceea sistemul bazat pe incredere este de preferat pentru ca putem alege ca alarma sa se declanseze doar la un nivel ridicat de exemplu 0.8 sau 80%

De asemenea un astfel de sistem nu poate fi utilizat nici pe o camera mobila pentru ca poate confunda miscarea fundalului cu miscarea unui obiect.

## Detaliere pe metode

color\_area(video\_frame, video\_base\_level=None)

In aceasta metoda folosim cele 2 culori care delimiteaza zona de hue in care apare focul.

Din moment ce exista posibiliatea de a capta mediul inconjurator in aceasta delimitare folosim numpy pe primul cadru al videoului pentru a obine culoarea medie care reiese din cadrul camerei. Nivelele de baza de verde si rosu sunt ajustate dupa nevoie ca sa nu existe ambiguitate. Desigur, un video care isi modifica des fundalul face calibrarea inutila, ca urmare putem utiliza si valorile de baza gata introduse.

Separam apoi zona de interes din spectrul HSV si adaugam thresholding pentru flacari albe care nu pot fi detectate de spectrul galbui-portocaliu. Aceasta modificare este optionala deoarece face sistemul vulnerabil la reflexii si surse de lumina puternice cum ar fi farurile unei masini.

area\_processing(local\_mask):

In aceasta funtie erodam filtrarea anterioara si apoi facem o masca cu thresholding dupa care dilatam produsul obtinut. Aceasta masca este perfecta pentru contour detection.

Restul de procesare este scrisa direct in metoda principala.

resized\_frame = cv.resize(frame, (640, 360), cv.INTER\_AREA)  
  
area = color\_area(resized\_frame, average)  
  
# Get Colored Area Ratio for fire colored pixel percentage  
ratio = cv.countNonZero(area) / (resized\_frame.size / 3)  
ratio \*= 100

Folosim o singura marime standard de 640x360 pentru convenienta si apoi obinem procentajul de pixeli care indeplinesc conditia vizuala initiala de flacara. In cazul in care depasim marja de 2,5% continuam apoi procesarea de contur.

FlameFluctuation = FlameSize - PastFlameSize  
FlameGrowth = FlameSize / PastFlameSize

Fluctuatia mica ne arata ca singurele schimbari din imagine sunt artefacte sau miscari laterale care nu schimba perspectiva / indica faptul ca flacara nu este in miscare. O deviatie de 160 de pixeli in cadrul unei imagini de 230,400 este acceptabila. Prin urmare daca in ultimele 5 cadre nu avem fluctuatii minore nu avem o flacara vie.

FlameGrowth ne arata miscarile mari ale focului prin procente de crestere si scadere.

Combinam cele 2 si avem posibilitatea reala a unui foc.

Folosind apoi o medie aritmetica pe ultimele 50 de cadre aflam nivelul de incredere.

## Bibliografie

https://scialert.net/fulltext/?doi=itj.2010.899.908 -- Fire Color Data & General Application

https://realpython.com/python-opencv-color-spaces/ -- Color Segmentation

https://stackoverflow.com/questions/66757199/color-percentage-in-image-for-python-using-opencv -- Image Color Percentage

https://docs.opencv.org/3.4/d3/db4/tutorial\_py\_watershed.html -- WaterShed Method for Segmentation

https://stackoverflow.com/questions/60863162/how-to-remove-small-objects-from-image-after-segmentation -- Removing contours by size

https://stackoverflow.com/questions/43111029/how-to-find-the-average-colour-of-an-image-in-python-with-opencv -- Average Color Level

https://www.geeksforgeeks.org -- Practical examples on stuff