



---

# Raport final - Supravegherea traficului-numărarea mașinilor

---

**ECHIPĂ: E10**

**Gîza David-Noel**  
Grupa 1309B

**Moroșanu Radu-George**  
Grupa 1309B

## 1 Rezumat

Procesarea imaginilor este domeniul în care imaginile sunt procesate de către o mașină (ex: calculator) pentru a extrage diferite informații pentru diferite domenii. La baza procesării imaginilor stau algoritmi specifici dar și tehnici matematice, o imagine putând fi privită ca o funcție de două variabile continue (de exemplu  $f(x,y)$ ).

Numărarea mașinilor asigură informații sigure despre fluxul traficului, accidente produse în trafic și posibilități de a reduce aglomerația în orele de vârf. Utilizarea procesării de imagini digitale este una dintre metodele cele mai eficiente pentru a obține informațiile optime necesare rezolvării acestor probleme. Prin acest proiect am încercat să observăm cum am putea îmbunătăți traficul, prin numărarea mașinilor.

Proiectul nostru constă în numărarea mașinilor ce traversează o anumită intersecție într-un anumit interval de timp. Am decis să folosim un video asupra căruia am aplicat operațiile necesare realizării acestei sarcini.

Am folosit obiecte de tip "BLOB" (Binary Large Object) acestea fiind obiecte binare de tip audio, video sau multimedia.

Folosind limbajul C++ și biblioteca OpenCV 4.5, am încărcat un video (.mp4). Video-ul reprezintă o camera de supraveghere asupra

unui sens de mers de pe o autostrada. Astfel, fiecare masina(obiect) reprezinta un BLOB. Masinile sunt urmarite cu ajutorul unui algoritm de predictie a pozitiei, apoi sunt detectate realizandu-se operatii specifice libreriei OpenCV, cum ar fi:segmentare, detectie contur si alte operatii morfologice:dilate,erode.

Totodata, am implementat o linie virtuala numita detector, care ajuta la numararea si detectarea masinilor. In functia de detectie aceasta este folosita drept bariera intre cadrul anterior si cel curent. Acest detector ne ajuta sa obtinem numarul final de masini.

**Cuvinte cheie: BLOB, segmentare, algoritm de predictie a pozitiei**

## 2 Introducere

Limbajul de programare ales este C++, totodata folosindu-ne de facilitatile libreriei OpenCV. Ca mediu de dezvoltare am folosit Visual Studio 2019, deoarece, suntem, in cea mai mare masura, familiarizati cu acesta.

Proiectul poate fi utilizat de companii care se ocupa de montarea sistemelor de supraveghere/monitorizare a traficului. Astfel, la anumite perioade de timp(ex:ore de varf in trafic) se poate evita aglomeratia pentru a imbunatati circulatia in zonele cu trafic ridicat.

Ne dorim ca pe parcurs sa imbunatatim acest proiect si sa avem o performanta cat mai ridicata, prelucrarea imaginilor fiind un domeniu care ofera multe avantaje, atat in prezent cat si in viitor.

## 3 Metode existente

Pentru a detecta, prezice pozitia si numara BLOB-urile am folosit urmatoarele metode:

1. Threshold(segmentare)-folosita pentru a separa obiectele de background.
2. ConvexHulls-incadreaza obiectele in forme convexe, metoda gasind poligonul convex a unui set de puncte 2D.
3. DrawAndShowContours-Functia traseaza conturul imaginii daca grosimea $\geq 0$  sau umple aria incadrata de contur daca grosimea $< 0$ .

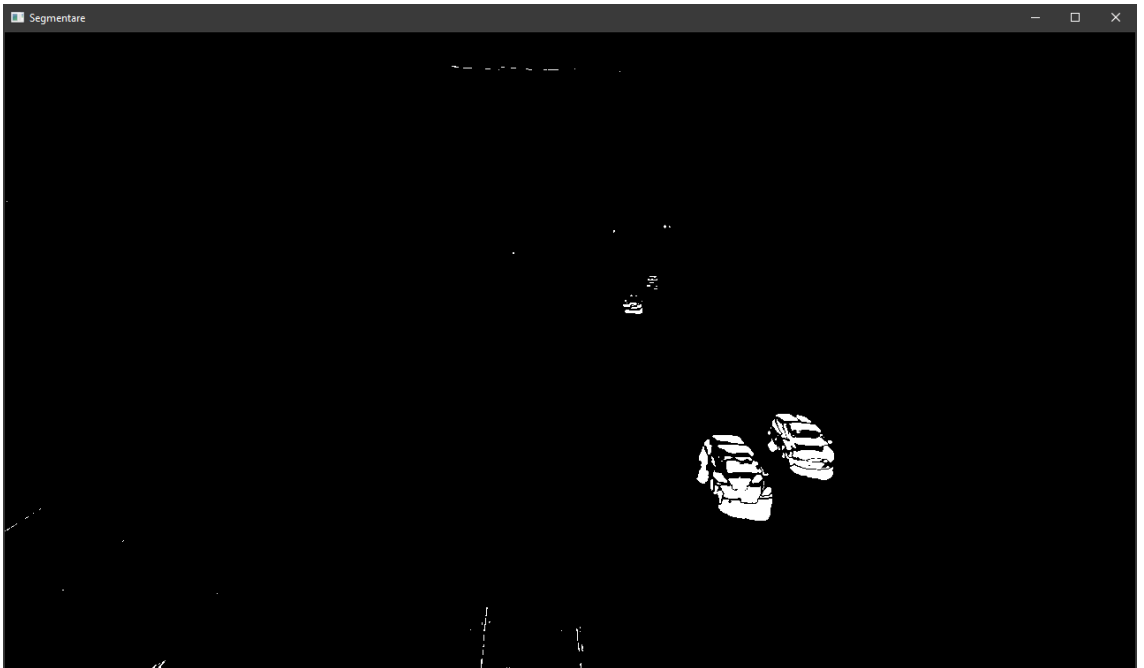


Figure 1: Segmentare

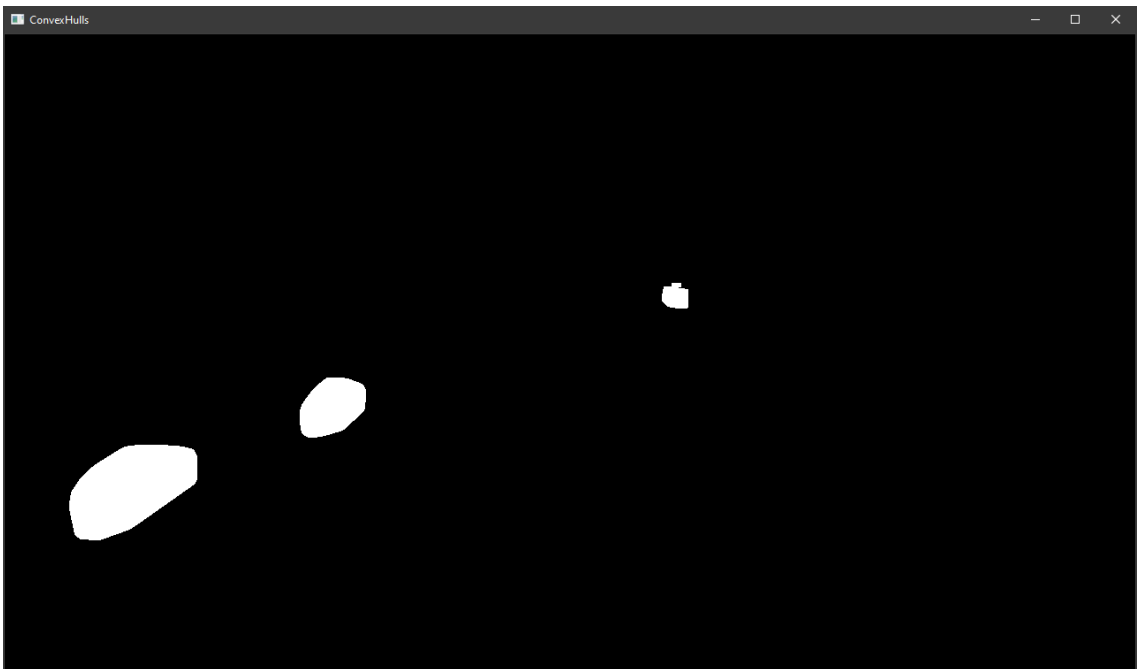


Figure 2: ConvexHulls



Figure 3: Contururi

#### 4 Descrierea tehnică a soluției

Am decis ca mai intai sa incarcam un video asupra caruia am realizat operatiile necesare numararii masinilor. Pentru incarcarea video-ului si tratarea erorilor am folosit urmatoarea functie:

```
VideoCapture capVideo;
capVideo.open("HighwayIntersection.mp4");

if (!capVideo.isOpened()) {
    std::cout << "Eroare la citire video" << std::endl << std::endl;
    _getch();
    return (0);
}

if (capVideo.get(cv::CAP_PROP_FRAME_COUNT) < 2) {
    std::cout << "Eroare: video-ul trebuie sa aiba cel putin 2 cadre";
    _getch();
    return (0);
}
```

Se intra in while loop-ul principal atat timp cat video-ul este deschis(ruleaza). Se lucreaza cu 2 cadre care se transforma din RGB in grayscale. Dupa am aplicat filtrul Gaussian pentru eliminarea zgomotului gaussian, iar apoi am aplicat segmentarea pentru a separa obiectele de background. Mai apoi, am creat o matrice tip filtru 3x3 care ajuta la aplicarea unor operatii morfologice: dilatare, eroziune. Am mai apelat functiile de gasirea conturului si afisarea contu-

rului pentru obiectele de tip point. Se construiesc vectorul de puncte convex hulls aplicand functia ConvexHulls. Se impun cateva conditii legate de dimensiunea blob-urilor apoi se face aplicarea efectiva a trecerii obiectului peste detector(linia de trecere), dupa care se afiseaza numarul de obiecte la momentul curent.

Alte functii folosite in cadrul proiectului sunt:

1. verificare\_trecere\_linie: Prin aceasta functie, se verifica daca blob-ul este inca detectat si daca acesta trece de suprafata pe care linia de verificare(detectorul) este amplasata, atunc blob-ul este numarat, astfel obtinandu-se numarul final de masini.

```
bool verificare_trecere_linie(
    std::vector<Blob>& blobs, int &poz_linie_orizontala, int&nr_masini) {
    bool OK = false;
    for (auto blob : blobs) {

        if (blob.still_tracked == true && blob.centerPositions.size() >= 2) {
            int prevFrameIndex = (int)blob.centerPositions.size()-2;
            int currFrameIndex = (int)blob.centerPositions.size()-1;

            if (
                blob.centerPositions[prevFrameIndex].y <= poz_linie_orizontala &&
                blob.centerPositions[currFrameIndex].y > poz_linie_orizontala) {
                    nr_masini++;
                    OK = true;
            }
        }
    }

    return OK;
}
```

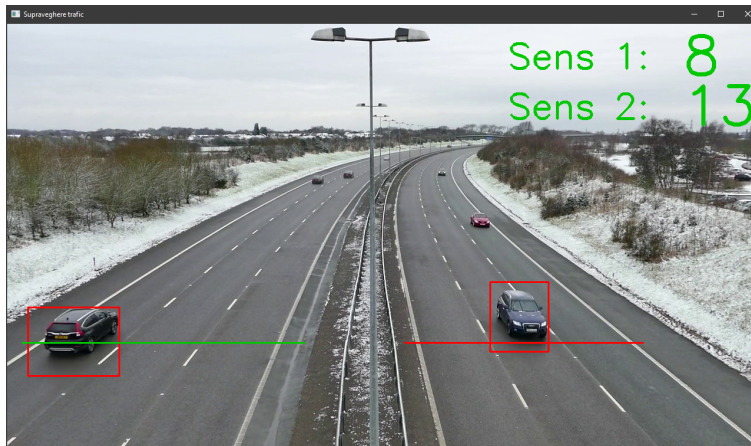


Figure 4: Imagine de la runtime

Pentru implementare exista 3 fisiere "Main.cpp", "Blob.cpp" si "Blob.h". In primul dintre ele se gasesc functiile necesare rezolvarii proiectului. In "Blob.h" este implemen-

tata clasa Blob si in "Blob.cpp" este implementat constructorul clasei si algoritmul de prezicere a pozitiei urmatoare.

Github: <https://github.com/davidgiza5/PI-P>

## **5 Rezultate experimentale**

In urma testarii si rularii programului, rezultatele obinute au o acuratete de aproximativ 85%. Am observat ca rezultatele difera in functie de anumiti parametri cum ar fi: pozitia detectorului, lungimea acestuia (daca este distribuit asupra video-ului in totalitate sau nu), conditiile de dimensiune a blob-urilor (aria in care sunt incadrate obiectele), numarul de cadre pentru care obiectul este inca detectat. O diferenta in numaratoare consta si in faptul ca la anumite cadre exista posibilitatea ca anumite masini care traverseaza in sens opus sa fie numarate de mai multe ori, astfel acuratetea fiind mai scazuta. De asemenea, am observat ca rezultatele difera si atunci cand un numar mare de masini traverseaza simultan, numaratoarea fiind gresita.

## **6 Concluzii**

In urma implementarii proiectului am ajuns la concluzia ca proiectarea imaginilor si lucrul cu imagini, este un domeniu de interes pentru noi, studentii, putand urma o cariera din acest domeniu si din lucrurile pe care le-am invatat. Proiectul ne-a ajutat sa ne aprofundam cunostintele in legatura cu facilitatile puse la dispozitie de biblioteca OpenCV. Fiind unul dintre primele noastre proiecte ca studenti, am realizat ca aceste tipuri de activitati sunt benefice pentru a ne pregati pentru examenul de licenta si totodata, pe viitor, sa stim cum trebuie sa ne organizam timpul si sarcinile la un eventual loc de munca.

Procesarea imaginilor este un domeniu in care studentii ce urmeaza acest curs, isi pot dezvolta si urma cariere.

Procesarea imaginilor medicale este una dintre cele mai intalnite cariere in care se utilizeaza lucrul cu imagini, domeniile in care intalnim aceasta activitate fiind:

1. Microscopia digitala (segmentare nucleu-celule, segmentare celule grasime pentru analiza steatozei hepatice).
2. segmentare tumori cerebrale, vase de sange, oase etc.

Alte domenii in care se utilizeaza procesarea imaginilor sunt:

1. Monitorizare, supraveghere: supravegherea traficului, detectia persoanelor(motion detection), detectia/recunoasterea fetelor etc.
2. Roboti mobili autonomi: detectie pietoni, detectie benzi circulatie, manipulare obiecte etc.

## Referințe

[https://www.researchgate.net/publication/308605824\\_Vehicle\\_counting\\_method\\_based\\_on\\_digital\\_image\\_processing\\_algorithms](https://www.researchgate.net/publication/308605824_Vehicle_counting_method_based_on_digital_image_processing_algorithms)

<https://www.sciencedirect.com/topics/engineering/image-processing>

<https://docs.opencv.org/3.4/index.html>

[www.mdpi.com/T1/guilsinglrighpdf](http://www.mdpi.com/T1/guilsinglrighpdf)