UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

Detectia benzilor de circulatie

Deac Mihai
grupa 302310

Cuprins

1. Rezumat:

Acest proiect propune implementarea unui algoritm eficient de detectare a benzilor de circulatie in timp real si determinarea deplasamentului de la centrul drumului. Imaginile sunt preluate dintr-un video, iar prelucrarea se face pe fiecare frame in parte aplicand o filtrare a liniilor albe, o transformare din planul strazii (perspectiva) in planul camerei (birdEye), o detectie de muchii, gasirea muchiilor corespunzatoare liniilor albe si apoi transformarea imaginii inapoi in planul strazii.

2. Introducere:

2.1 Contextul temei

Prelucrarea imaginilor este un domentiu tot mai extins folosit pe scara foarte larga pentru extragerea anumitor informatii din mediul inconjurator. Probabil cel mai intalnit domeniu in care se foloseste procesare de imagine este automotive pentru determinarea pozitionarii masinii in trafic. De aceea am ales dezvoltarea unui sistem capabil sa detecteze benile de circulatie.

2.2 Definirea problemei

Se observa o necesitate foarte mare pentru generatia de masini autonome sa poata determina pozitia lor in trafic si mai exact pozitia relativa la banda de curculatie. Astfel, pentru identificarea acesti pozitii este necesara recunoasterea celor doua linii ce incadreaza banda de circulatie si determinarea unui unghi si a unui deplasament de la centrul bezii.

2.3 Solutii propuse

Exista mai multe modalitati de detectie a liniilor albe. In primul rand este necesara separarea lor de restul imaginii acest lucru fiind in efectuat in prima etapa prin urmarirea comportamentului culorii intr-o anumita gama. Astfel se observa ca in gama HSL (Hule, Saturation, Lightness) componenta de luminozitate a culorii alb are o valoare ridicata comparativ cu valoarea ei in celelalte doua canale. Utilizand o astfel de selectie se poate filtra liniile dupa urmatoarea formula:

$$lightness(pixel) > hue(pixel) + saturation(pixel)$$

Rezultatul obtinut este urmatorul:

In continuare exista mai multe metode de abordare a detectiei de linii: fie direct pe aceasta imagine aplicand functia Canny de detectie a muchiilor si apoi selectarea relevante, fie pe imaginea translatata in vizualizarea bird-eye (proiectie pe verticala). In acest proiect a fost abordata metoda de proiectare a planului strazii pe verticala. Astfel, se selecteaza patru puncte din planul drumului si patru puncte ce vor reprezenta colturile noii imagini. Cu aceste coordonate se calculeaza matricea de homografie. Dupa ce avem aceasta matrice putem sa o aplicam asupra fiecarui punc din regiunea de interes a imaginii sursa pentru a determina noua coordonata in imaginea destinatie.
Avand acum o privire de sus asupra benzii de circulatie se poate aplica un algoritm de scoatere in evidenta a muchiilor cum este Canny care calculeaza modulul operatorilor sobel si gradientul imaginii, dupa care efectueaza o binarizare cu un prag stabilit.
Pentru a determina doar muchiile care pot candida pentru a marcaje stradale se poate aplica unul din urmatorii algoritmi:

RANSAC (Random Sampla Consensus)

- Aceasta functie gaseste linia care are cele mai multe puncte in vecinatatea sa la o distanta minima stabilita.
- Se aleg aleator doua puncte de muchie care vor determina o dreapta si se calculeaza distanta de la toate celelalte puncte de muchie la aceasta dreapta. Apoi sunt contorizate doar punctele care sunt la o distanta mai mare decat o valoare prestabilita
- Se repeta alegerea celor doua puncte pana cand s-au epuizat toate variantele sau pana cand gasim o dreapta care aproximeaza un numar acceptabil de puncte
- Se retine dreapta cu cel mai mare numar de puncte aproximate
- Acest algoritm nu este eficient din punct de vedere al timpului de executie si nu ste nici consistent, timpul variind in functie de punctele alese

Transformarea Hough

- Transformarea Hough determina cate puncte contine fiecare dreapta din imagine.
- Pentru fiecare punct de muchie din imagine se calculeaza 2 parametrii: unghiul tetha reprezentat in acest caz de gradient-ul calculat anterior si r determinat de ecuatia parametrica a dreptei:

$$r = x\cos\theta + y\sin\theta \tag{1}$$

- Aceste doua valori vor reprezenta parametrii unei drepte in planul de coordonate tetha, r.
- Fiecare doua linii care se intersecteaza in planul tetha, r reprezinta un doua puncte de pe aceeasi linie in planul imaginii (x,y).
- Acest algoritm este mai eficient necesitand doar parcurgerea tuturor punctelor de muchie o singura data

Dupa aplicarea transformatei Hough se aleg cele doua linii care contin cele mai multe puncte, acestea fiind liniile benzii de circulatie si se returneaza 4 puncte (cate 2 de pe fiecare linie) pentru a putea fi transformate inapoi in planul strazii.

3. Fundamentare teoretica:

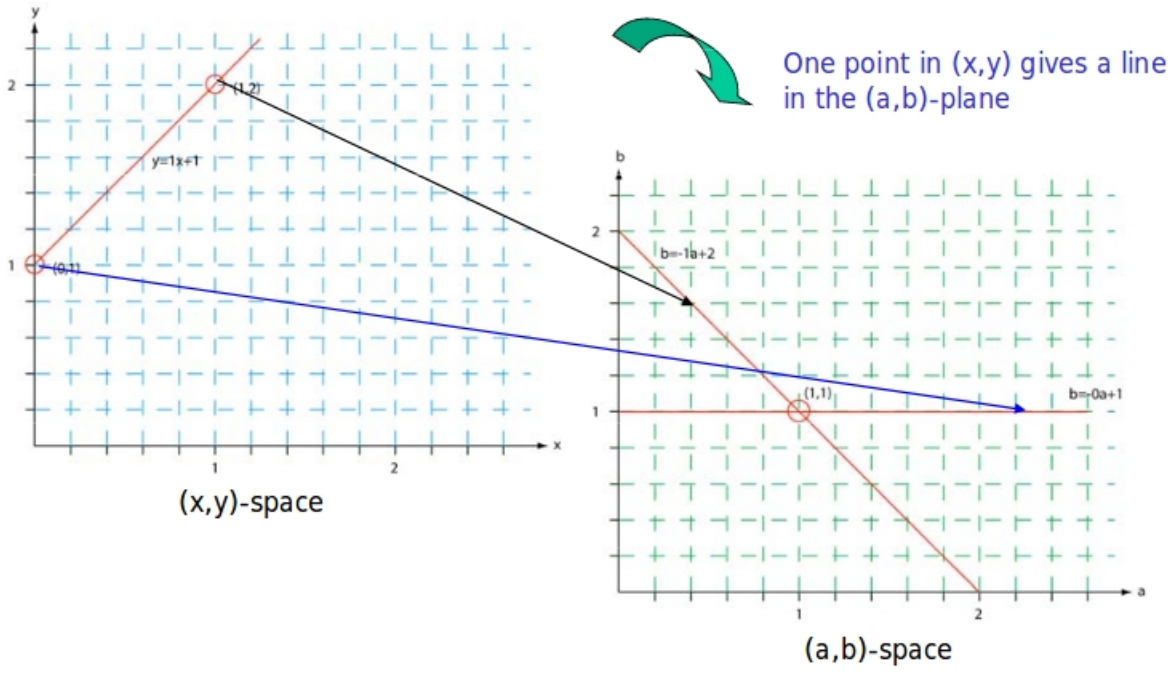
Translatarea punctelor imaginii dintr-un plan in altul:

$$\begin{bmatrix} x'/\lambda \\ y'/\lambda \\ \lambda \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2}$$

$$PH = \begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1x'_1 & y_1x'_1 & x'_1 \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1y'_1 & y_1y'_1 & y'_1 \\ -x_2 & -y_2 & -1 & 0 & 0 & 0 & x_2x'_2 & y_2x'_2 & x'_2 \\ 0 & 0 & 0 & -x_2 & -y_2 & -1 & x_2y'_2 & y_2y'_2 & y'_2 \\ -x_3 & -y_3 & -1 & 0 & 0 & 0 & x_3x'_3 & y_3x'_3 & x'_3 \\ 0 & 0 & 0 & -x_3 & -y_3 & -1 & x_3y'_3 & y_3y'_3 & y'_3 \\ -x_4 & -y_4 & -1 & 0 & 0 & 0 & x_4x'_4 & y_4x'_4 & x'_4 \\ 0 & 0 & 0 & -x_4 & -y_4 & -1 & x_4y'_4 & y_4y'_4 & y'_4 \end{bmatrix} \begin{bmatrix} h1 \\ h2 \\ h3 \\ h4 \\ h5 \\ h6 \\ h7 \\ h8 \\ h9 \end{bmatrix} = 0 \tag{3}$$

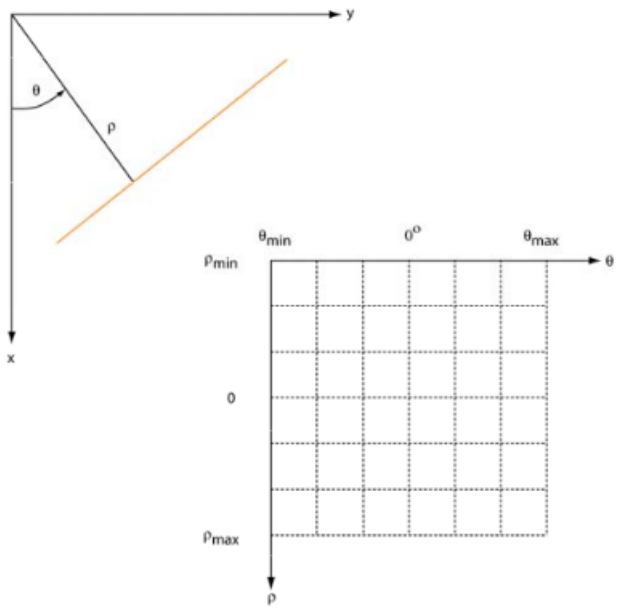
Transformarea Hough

$$b = -xa + y \tag{4}$$



- Cele doua puncte (x,y) si (z,k) definesc o linie in planul (x,y)
- Acete puncte determina doua drepte diferite in spatiul (a,b) care se intersecteaza
- Astfel, toate punctele de pa linia definita de (x,y) si (z,k) vor parametriza drepte care se vor intersecta in sistemul (a,b)
- Punctele care se afla pe aceeasi dreapta vor forma un “cluster of crossings” in spatiul (a,b)

$$x\cos\theta + y\sin\theta = \rho \tag{5}$$



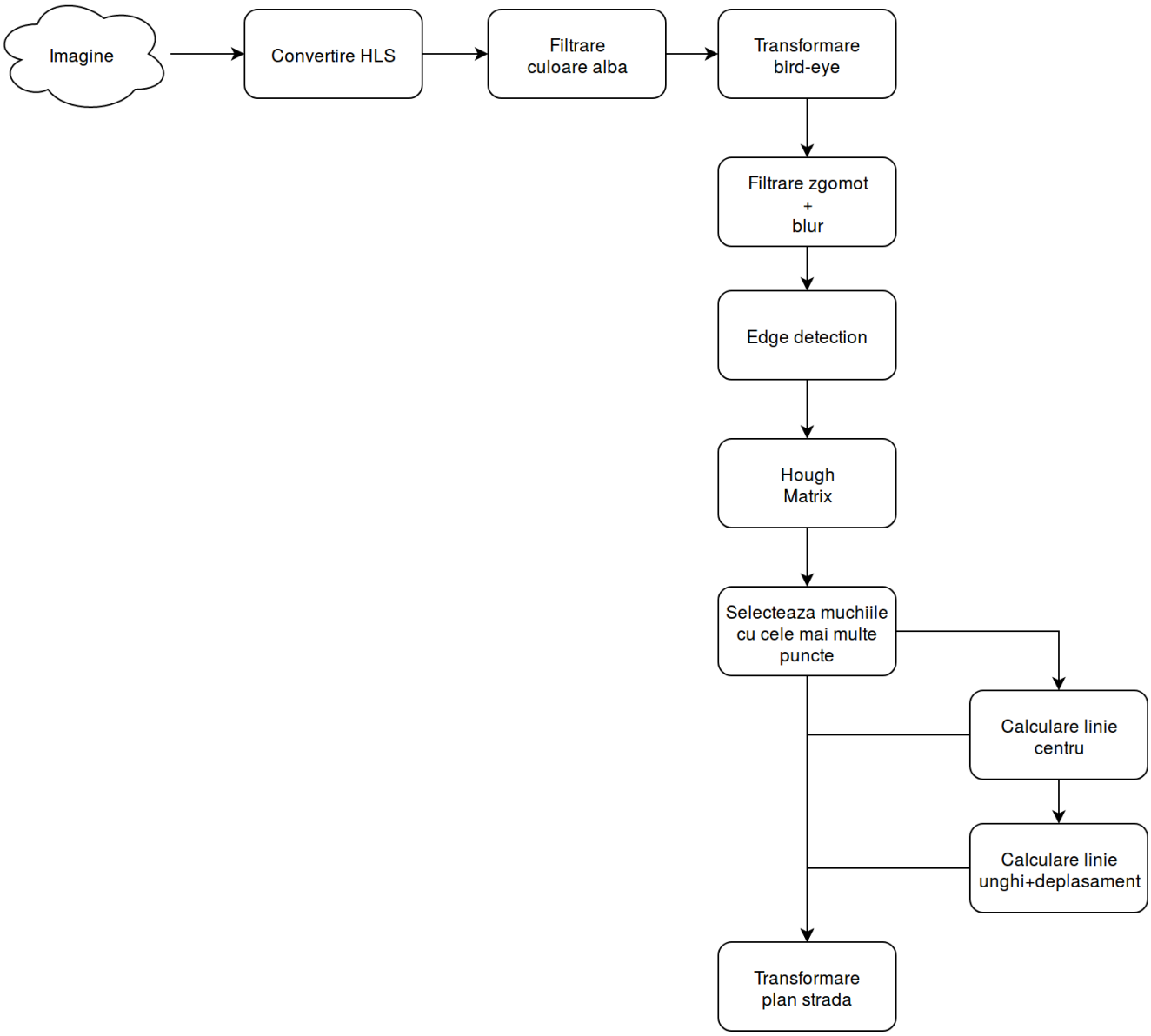
Polar representation of lines

Intervalul in care θ poate lua valori este 0 – 180 grade, iar intervalul in care ρ poate sa ia valori este 0 – N*sqrt(2)

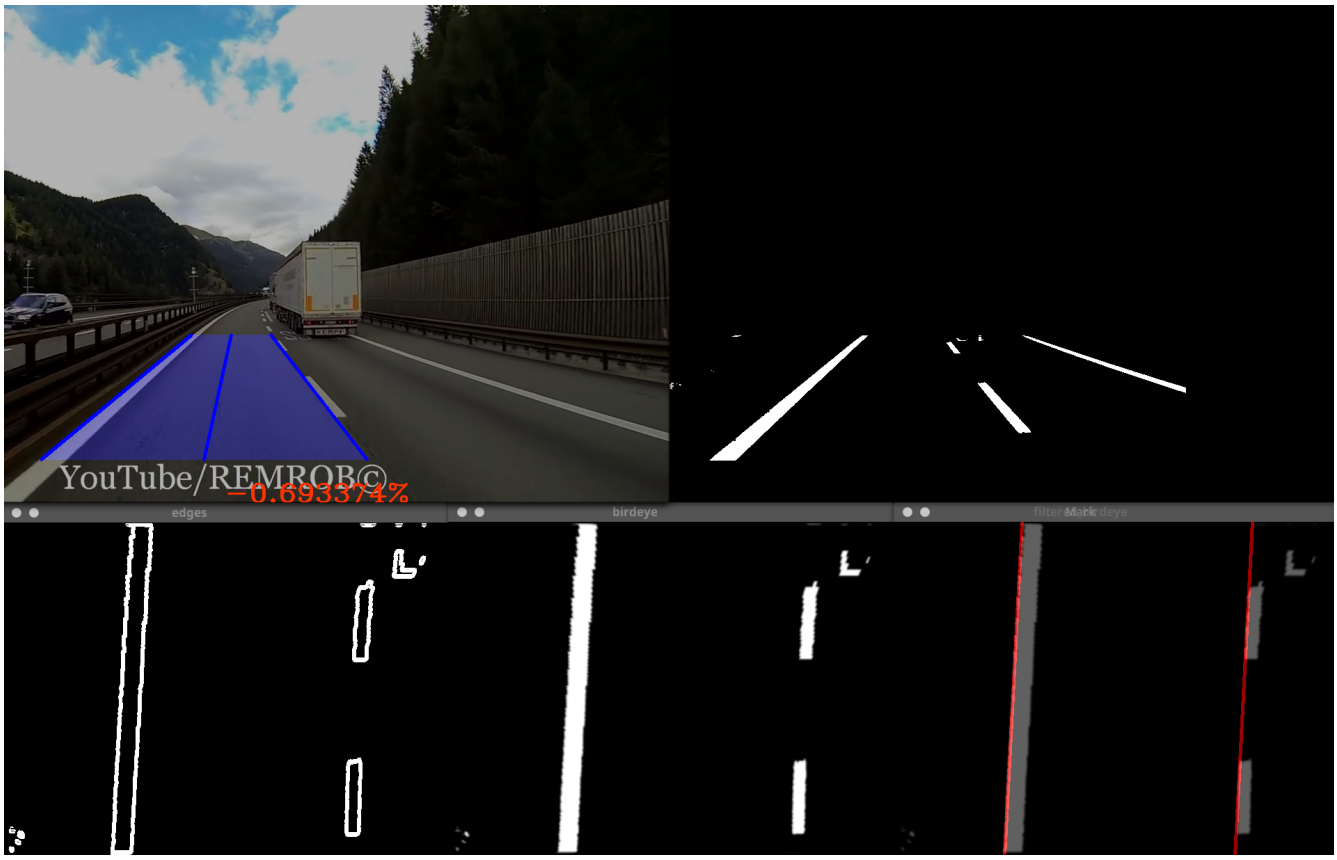
4. Proiectare si implementare:

Implementarea a fost realizata utilizand libraria openCV in limbajul de programare C++.
Au fost implementate urmatoarele functii:

- `Mat getHomographyMatrix(Point2f *src_vertices, Point2f *dst_vertices)`
  - determina cele 9 valori ale matricii de homohrafie pornind de la cei doi vectori de puncte de intrare utilizand metoda SVD::zolveZ pentru rezilvarea sistemului maticial (3)
- `void perspectiveWrap(Mat perspective_image, uchar *orthogonal, float* homographyMatrix, int width, int height)`
  - aplica transformata de perspectiva pe imaginea sursa utilizand matricea de homografie
  - se inmulteste fiecare coordonata a regiunii de interes a imaginii cu matricea de transformare pentru a obtine noile coordonate conform (2)
- `void lineFilter(Mat src, uchar* dst, int width)`
  - converteste imaginea in gama te culoare HSL utilizand functia din openCV
  - separa cele 3 canale ale imaginii si compara luminozitatea cu suma celorlalte 2 canale
- `vector<Point2d> edgesDetection(uchar* src, uchar* dst, int width, int height, int min_line, int max_line, int threshold)`
  - aplica cei doi operatori Sobel pentru pronuntarea muchiilor si calculeaza modulul acestora
  - calculeaza gradientul
  - calculeaza matricea de homografie deoarece este mai eficient sa fie calculata in acelasi ciclu repetitiv cu gradientul. Astfel incrementeaza in matricea H valoarea de la pozitia data de [tetha, p]
  - returneaza cele 4 puncte pentru cele 2 linii de ccirculatie
- `void perspectivePoints(Point2d* src_points, Point2d* dst_points, float* homographyMatrix, int pointsNo)`
  - calculeaza punctele transformate in alt plan utilizand maticea de homografie
- `vector<Point2d> houghPoints(uchar* src, Mat hough, int width, int height, int min_line, int max_line, bool limited_lines=true)`
  - determina cele 4 puncte de liniilor ce incadreaza banda de circulatie cautand linia ce contine cele mai multe puncte in partea stanga a imaginii si linia ce contine cele mai multe puncte in partea dreapta a imaginii


Imagine
Convertire HLS
Filtrare
culoare alba
Transformare
bird-eye
Filtrare zgomot
+
blur
Edge detection
Hough
Matrix
Selecteaza muchiile
cu cele mai multe
puncte
Calculare linie
centru
Calculare linie
unghi+deplasament
Transformare
plan strada

5. Rezultate experimentale



Bibliografie

https://www.uio.no/studier/emner/matnat/ifi/INF4300/h09/undervisningsmateriale/hough09.pdf
http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm
http://6.869.csail.mit.edu/fa12/lectures/lecture13ransac/lecture13ransac.pdf
http://www.uio.no/studier/emner/matnat/its/UNIK4690/v16/forelesninger/lecture_4_3-estimating-homographies-from-feature-correspondences.pdf
http://portal.uc3m.es/portal/page/portal/dpto_ing_sistemas_automatica/investigacion/lab_sist_inteligentes_old/publications/iv05-b.pdf
https://towardsdatascience.com/finding-lane-lines-on-the-road-30cf016a1165