**Detectarea linei orizontului**

**Git**: <https://github.com/MihaiOlah/Horizon-Line-Detection>

**Descriere generală**

Programul încearcă să găsească linia orizontului după definiţia acesteia din contextul vederii artificiale pentru vehicule. Pentru aceasta se încearcă detectarea unor linii paralele, care converg într-un punct (vanishing point). Găsirea acestui punct va duce automat şi la găsirea liniei orizontului, fiindcă aceasta îl intersectează.

Pentru detectarea liniilor paralele, algoritmul va încerca detectarea liniilor de pe marginea drumului. De obicei, acestea au o culoare galbenă sau albă. Pentru acestea s-au definit nişte intervale de culoare, fiindcă dacă sunt albe pot fi murdare, deci nuanţe de gri deschis sau de nuanţe diferite de galben. Se va face o imagine temporară, unde toate punctele care se încadrează în acele intervale vor fi marcate cu alb, iar cele care nu se încadrează sunt negre. Astfel, imaginea de lucru va avea doar puncte albe şi negre, unde punctele de interes vor fi cele albe.

Din cauza faptului că poate apărea zgomot în imagine, de exemplu alte obiecte albe sau galbene, se va face o filtrare, unde dacă un pixel alb nu este înconjurat de un număr de alţi pixeli albi pe o anumită arie va fi marcat şi el cu negru, fiind considerat zgomot.

Folosind poza descrisă mai sus, care conţine doar pixeli albi şi negri, se vor căuta ecuaţiile dreptelor care trec prin cele mai multe puncte albe. Într-o listă vom păstra tupluri care conţin numărul de puncte unite, panta şi constanta dreptei. Deoarece punctele s-ar putea să nu se potrivească perfect, vom lăsa o toleranţă pentru care un punct este considerat încadrat pe dreaptă.

Din motive de eficienţă şi de eliminare de zgomot, scanarea se va face pentru punctele din jumătatea inferioară a imaginii (considerăm că cea de sus este cerul), două treimi la stânga şi două treimi la dreapta, pentru a compensa pentru eventuala poziţionare a maşinii pe bandă.

Din lista de drepte, se vor alege top 5 cele mai aproapiate drepte din scanarea de la stânga şi se va face o dreaptă rezultantă medie şi la fel se va face o dreaptă rezultantă pentru scanarea de la dreapta. Aceste două drepte se vor considera paralele şi se va căuta punctul lor de intersecţie. Acesta va fi vanishing point-ul, prin care va fi trasată linia orizontului.

Pentru imaginile mari, căutarea este lentă, dar s-a adăugat o opţiune care micşorează poza pentru a obţine o imagine mai uşor de prelucrat. Avantajul este că se va face mult mai rapid, dar rezultatul nu va mai fi la fel de precis.

**Explicarea argumentelor**

Programul ia două argumente, unul obligatoriu care va conţine calea spre imagine şi unul opţional care să indice dacă se foloseşte versiuinea rapidă a algoritmului.

* '-p'/'--imagePath': va conţine calea spre imaginea care urmează să fie prelucrată.
* '-s'/'--short': argument opţional boolean, implicit este False, care va indica folosirea versiunii lente, dar mai precise a programului. Dacă este dat, atunci va fi True şi va fi folosită cealaltă variantă.

**Explicarea constantelor**

MAX\_WHITE = (255, 255, 255)

MIN\_WHITE = (180, 180, 170)

MIN\_YELLOW = (75, 120, 170)

MAX\_YELLOW = (105, 180, 200)

Constantele MIN\_WHITE, MAX\_WHITE, MIN\_YELLOW şi MAX\_YELLOW definesc capetele intervalelor pentru culorile galben şi alb.

NEIGHBOUR\_MATCH\_PERCENTAGE = 0.25

NEIGHBOUR\_MATCH\_WIDTH = 2

NEIGHBOUR\_MATCH\_PERCENTAGE reprezintă procentul de vecini din aria definită de NEIGHBOUR\_MATCH\_WIDTH pentru un pixel alb care să fie şi ei albi pentru a nu fi considerat zgomot.

SLOPE\_ANGLE\_START = 20

SLOPE\_ANGLE\_STOP = 70

SLOPE\_ANGLE\_START şi SLOPE\_ANGLE\_STOP reprezintă pantele de la care începem căutarea şi până la care pantă maximă mergem. Nu are rost să luăm în considerare drepte prea orizontale, fiindcă ne aşteptăm ca linia orizontului să nu fie în partea cea mai de jos a imaginii, şi nici prea verticale, fiindcă atunci s-ar intersecta afară din imagine.

Y\_TOLERANCE = 2

Y\_TOLERANCE reprezintă toleranţa pentru coordonatele unui punct ca să fie considerat că este încadrat pe o dreapta. În acest caz reprezintă doi pixeli în sus pe axa OY şi doi în jos faţă de pixelul ţintă.

**Explicarea funţiilor**

* parse\_args(): Prelucrează argumentele date din linia de comandă.
* is\_in\_color\_range(color): Verifică dacă pixelul color este încadrat în vreun interval de culoare (alb sau galben).
* convert\_to\_yellow\_and\_white(img): Coverteşte imaginea în imaginea utilitară formată din pixeli albi şi negri.
* check\_neighbours(img, row, column): Verifică dacă un pixel din imagine la coordonatele date are suficienţi vecini pentru a nu fi considerat zgomot.
* filter\_by\_neighbour(img): Parcurge fiecare pixel şi elimină pixelii albi care nu îndeplinesc criteriul cu numărul de vecini.
* find\_points\_for\_line(img, slope, constant, y\_tolerance), find\_points\_for\_line2(img, slope, constant, y\_tolerance): Numără câte puncte din imagine se încradează pe ecuaţia dreptei.
* get\_constants(img, slope), get\_constants2(img, slope): Generează constantele pentru o ecuaţie a dreptei ştiind panta.
* find\_lines(img, slope\_angle\_start, slope\_angle\_stop), find\_lines2(img, slope\_angle\_start, slope\_angle\_stop): Formează toate ecuaţiile de drepte care urmează să fie verificate.
* draw\_line(img, result, y\_tolerance, thickness), draw\_line2(img, result, y\_tolerance, thickness): Desenează în imagine o dreaptă orizontală.
* make\_red\_line(image, x, y, thickness): Face o linie orizontală roşie.
* pyramid(img): Micşorează imaginea prin piramidare.
* shrink\_image(img): Aplică piramidarea până când are lăţimea şi înalţimea mai mici de 400 px.
* main(): Apelează funţiile pentru scanarea de la dreapta la stânga şi de la stânga la dreapta, reface scalarea, face intersecţia dreptelor rezultate şi desenează orizontul.
* cv2.imshow(): Afişarea imaginii.
* cv2.imread(): Citirea imaginii.

**Indicator de performanţă**

Testarea preciziei s-a făcut pe un set de 50 de poze. Mai jos e prezentat rezultatul atât pentru varianta mai lentă, cât şi pentru cea rapidă.

**Varianta lentă** un procentaj de reuşită de 52%. Imaginile corecte din setul de teste sunt: 0, 1, 3, 7, 9, 12, 14, 15, 17, 18, 20, 23, 27, 29, 30, 31, 33, 35, 42, 43, 44, 45, 46, 47, 48, 49.

**Varianta rapidă** un procentaj de reuşită de 30%. Imaginile corecte din setul de teste sunt: 0, 6, 7, 9, 15, 24, 25, 29, 30, 31, 43, 45, 46, 47, 49**.**

După cum se observă, înclinaţia drumului şi curbele au un impact mare pentru găsirea liniei orizontului aşa cum este definită.