Универзитет у Београду
Електротехнички факултет
Катедра за сигнале и системе

Милош Милошевић 3186/2016

Мастер рад

Препознавање разделних линија коловозних трака на основу снимка са камере на ветробранском стаклу

Ментор: Доц. др Вељко Папић

Београд
август 2018.

Захвалност

Желео бих да се захвалим професору Вељку Папићу на несебичној помоћи при изради овог рада, као и на енергији и знању које активно преноси на студенте. Желео бих да се захвалим и колективу Електротехничког факултета јер су мени и многим мојим колегама обезбедили драгоцено знање које нам је поплочало пут успеха на многим животним пољима.

На крају, највећу захвалност дугујем својој породици.

Сажетак

Циљ овог рада је развој алгоритма за препознавање разделних линија коловозних трака на основу видео снимка који се добија са камере постављене на ветробранско стакло аутомобила. Развијени алгоритам комбинује знања из области дигиталне обраде слике, препознавања облика и рачунарске технике. Програм, који примењује предложени алгоритам, написан је у C++ програмском језику, а за потребе дигиталне обраде слике користи позиве из рачунарске библиотеке *OpenCV.* Брзина обраде програма је довољно велика да се обрада може одвијати у реалном времену.

Кључне речи

Lane detection, Driver assistance, V*ideo processing, Computer vision, OpenCV*

Садржај

Листа симбола

ABS	*Anti-lock Breaking System*	- систем за контролу проклизавања
ADAS	*Advanced driver-assistance system*	- системи за напредну помоћ возачу
BGR	*Blue Green Red*	- систем боја
ESP	*Electronic Stability Program*	- систем за контролу стабилности
HSV	*Hue Saturation Value*	- систем боја
LIDAR	*LIght, Detection And Ranging*	- врста сензора
OpenCV	*Open Source Computer Vision*	- рачунарска библиотека за обраду слике
RGB	*Red Green Blue*	- систем боја
ROI	*Region Of Interest*	- област слике која је од значаја

1. Увод

Према подацима светске здравствене организације, 1,25 милиона људи годишње изгуби живот у саобраћајним незгодама, а скоро 50 милиона људи задобије тешке телесне повреде. Друмски саобраћај постаје један од најнебезбеднијих видова превоза. Најчешћи узрок несрећа је људска грешка, попут вожње великом брзином, вожње под дејством алкохола или недовољне присебности возача. [1]

Инжењери данас напорно раде на развоју системâ који ће помоћи возачу и смањити број саобраћајних незгода. Овај рад позабавиће се једним од таквих система, системом за препознавање разделних линија коловозних трака. Препознавањем разделних линија коловозних трака можемо добити информацију о релативном положају аутомобила унутар саобраћајне траке. Ова информација може бити од пресудног значаја за предупређивање нежељеног преласка аутомобила у другу траку.

Структура рада састоји се од осам целина. Након уводног поглавља дат је опис и архитектура тренутних система за напредну помоћ возачу. Треће поглавље описује дигиталну обраду слике коришћењем *OpenCV* рачунарске библиотеке. Друго и треће поглавље служе као теоријски увод за предложено решење које је описано у четвртом поглављу. У петом поглављу дата је имплементација предложеног решења, а у шестом је описано аутоматско тестирање предложеног решења. Претпоследње поглавље представља осврт на успешност алгоритма и даје предлоге за његово унапређење. Последње поглавље садржи списак литературе која је коришћена приликом реализације рада.

2. Системи за напредну помоћ возачу

Рачунарски системи за напредну помоћ возачу, познатији под акронимом ADAS (од енгл. *Advanced Driver Assistance Systems*), спадају у врсту уграђених рачунарских система (енгл. e*mbedded systems*) који имају примену у аутомобилској индустрији. Почели су да се развијају почетком двадесет и првог века, а данас су део стандардне опреме луксузнијих модела готово свих произвођача аутомобила. Главни циљ система за напредну помоћ возачу је повећање безбедности у саобраћају. Овај циљ остварује се паралелном обрадом сигнала са великог броја сензора. Обрада сигнала одвија се у реалном времену и као резултат обраде доноси се закључак о стању у непосредној околини аутомобила. У случају препознате опасности, ADAS упозорава возача. Имајући ово у виду, јасно је да алгоритми система за напредну помоћ возачу морају бити поуздани и ефикасни.

Практично гледано ADAS представља фасадни термин скупа подсистема развијених са циљем подизања безбедности и комфора у саобраћају. Неки од данас познатих ADAS подсистема имају могућности контоле проклизавања точкова, аутоматског прилагођавања крстареће брзине аутомобила, аутоматског паркирања, препознавања поспаности возача, избегавања чеоног судара, препознавања саобраћајних знакова, препознавање саобраћајних трака итд.

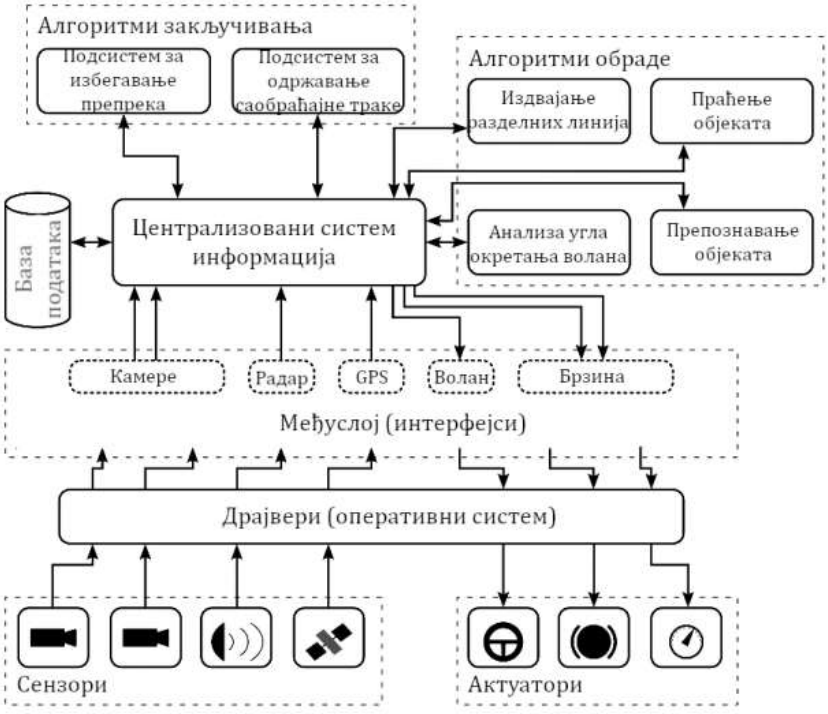
Скуп подсистема ADAS-а који су уграђени у аутомобил и њихова поузданост одређују ниво аутономости тог возила. Тренутно, дефинисано је пет нивоа аутономности возила. Строге законске регулативе прописују услове које модели аутомобила морају да испуне како би били сврстани у сваки од нивоа. Слика 1 приказује градацију нивоа аутономности. Највиши ниво аутономности постигнут у овом тренутку јесте ниво 4. [2]

Слика 1: Нивои аутономности возила

2.1. Архитектура система за напредну помоћ возачу

Нагли развој рачунарске науке и проналазак нових алгоритама за обраду информација резултовали су напретком подсистема за помоћ возачу. Данас се произвођачи утркују ко ће понудити поузданији и ефикаснији механизам за препознавање саобраћајних знакова, избегавање судара, контролу проклизавања итд. Међутим, уклопити све подсистеме у ефикасан и оптималан систем је постао озбиљан изазов. Отежавајућа околност у овом изазову је чињеница да су ADAS подсистеми по свом типу временски-критични наменски системи који се извршавају у реалном времену (енгл. *hard real-time system*) што значи да је време за које обраде улазну инфорамцију подједнако важно као и закључак о њој. Некада су ови подсистеми радили као независни системи, са независним изворима информација. Сензори једног од подсистема често су прикупљали исте податке које су прикупљали и сензори других подсистема, стварајући непотребну редудантност. Данас, потребан је другачији приступ. Дељење информација са сензора је нужно. Штавише, често су резултати обраде једног од подсистема улазне информације других подсистема.

Један од приступа који се намеће као решење поменутог изазова је модуларна архитектура са централизацијом информација. Увођењем још и додатног слоја обраде, који би био посредник између алгоритама за обраду информација и сензора и актуатора, постиже се далеко већа ефикасност од приступа у којем подсистеми функционишу као независне целине. Слика 2 даје графички приказ поменуте архитектуре.



Слика 2: Архитектура модерних система за напредну помоћ возачу

Предности овакве архитектуре су бројне. Најважнија предност је могућност независног тестирања сваког од модула, чиме се постиже знатно већи ниво сигурности. Модуларни развој омогућава писање тестова који се могу аутоматски покретати и који испитују модуле независно. Тиме се смањује вероватноћа грешке. Затим, увођењем међуслоја и интерфејса за размену података (слика 2) поставља се конвенција за размену података коју сви подсистеми прате. Постојање конвенције омогућава да исти подаци буду искоришћени у обради више алгоритама, чиме се смањује број потребних сензора и потребних канала за размену података. Постојање конвенције олакшава и размену података у ситуацији где је излазни податак једног од подсистема уједно и улазни податак другог подсистема. На самом крају, након сваке обраде, подаци се смештају у базу података. Историја података доступна је свим алгоритмима и понекад може одиграти пресудну улогу у одлучивању. [3]

Наравно, овако оптимална аритектура и идеја о доступности и дељењу информација не значи пуно уколико изостане подршка доброг оперативног система. Улога оперативног система је да што више алгоритама своју обраду заврши у предвиђеном року. Оперативни систем је срж временски-критичног наменског система. Он додељује ресурсе (процесор и меморију) алгоритмима за обраду по принципу приоритета и обавља синхронизацију процеса. Најчешће су у употреби оперативни системи који су развијени по угледу на *Linux* оперативни систем.

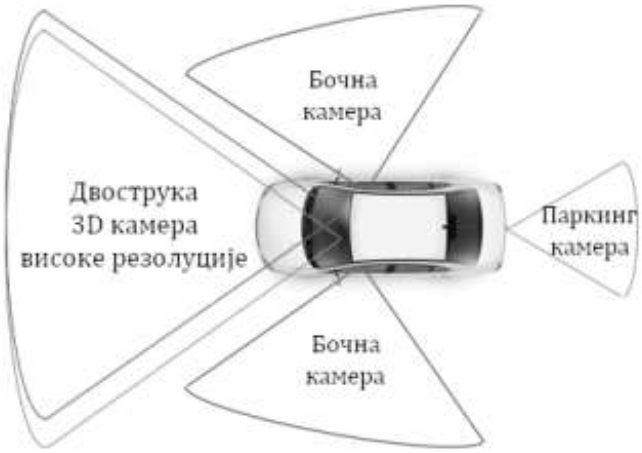
2.2. Сензори система за напредну помоћ возачу

Сензори играју важну улогу у сваком рачунарском систему. Информација која од њих стиже мора да одговара стању у реалном свету. У случају система за напредну помоћ возачу, тачност и поузданост информација мора бити на највишем нивоу. Имајући то у виду, јасно је да је пуно времена посвећено анализи које врсте сензора и које позиције у аутомобилу доносе најбоље резултате. Сензори који су део система за напредну помоћ возачу су акцелерометри, видео камере, радари, ултразвучни сензори итд. Напреднији системи се ослањају и на термовизијске камере.

Први системи који су развијени, а за које можемо рећи да обезбеђују помоћ возачу, радили су само при мањим брзинама. Такви системи су једноставни и имају домет од пар метара. Нуде помоћ при паркирању и праћењу „мртвог угла“. Идеалан избор сензора за поменуте сврхе су ултразвучни сензори. Они омогућавају детекцију објеката у непосредној близини аутомобила и мерење удаљености тих објеката од возила. Заснивају свој рад на одашиљању ултразвучних таласа и мерењу времена које је потребно да се талас одбије од оближњег објекта. Главна предност ултразвучних сензора је њихова мала цена, мали утрошак електричне енергије и неосетљивост на боје. Ултразвучни сензори се постављају на бранике и крила аутомобила дуж целог обима. Неретко их има и до двадесет.

Када се говори о сензорима система за помоћ возачу, јасно је да су видео камере прва асоцијација. Њихова улога је да процесорским јединицама обезбеде информације које су иначе доступне возачу путем његовог вида. Системи који укључују видео камере најчешће покушавају да алгоритамски опонашају закључивање које возач спроводи у својој глави.

Кључни проблеми приликом употребе видео камера јесу квалитет слике које оне обезбеђују и њихове позиције на аутомобилу. Камера мора бити прилагодљивија на промену амбијенталног осветљења, треба да услика квалитетну слику са довољно детаља, али што мањих димензија. Са растом резолуције слике расте и количина детаља на њој, али и потребни хардверски ресурси за обраду. Савремени аутомобил је опремљен са више камера које мотре на различиту околину аутомобила. На слици 3 приказан је најчешћи распоред камера. Најкорисније су две камере које су постављене ветробранско стакло и које омогућавају формирање 3D слике сцене испред аутомобила. Сигнал који је коришћен као извор информација у овом раду долази са једне камере постављене на ветробранско стакло.



Слика 3: Најчешћи распоред видео камера на аутомобилу

Наредни сензор у ADAS револуцији је LIDAR сензор (од енгл. *LIght, Detection And Ranging*). LIDAR сензор има улогу у мерењу растојања. Принцип рада му је веома сличан ултразвучном сензору. Разлика између ова два типа сензора је природа таласа који емитују. Ултразвучни сензор се ослања на ултразвучни талас, који је по природи механички талас, а LIDAR сензор користи ласерски сноп, који је по природи електромагнетни талас. Очита предност електромагнетних таласа наспрам механичких је њихова брзина, па LIDAR сензори имају далеко бржи одзив од ултразвучних сензора. Друга, можда и важнија, предност LIDAR сензора је њихов домет. У складу са тим, LIDAR сензори су нашли примену у праћењу саобраћаја на већим удаљеностима. Ова врста сензора је примарни извор информација за подсистеме задужене за контролу крстареће брзине и избегавање чеоних судара. LIDAR сензори се користе и за снимање топографије терена и морског дна.

Данас, најнапреднији модели поседују и термовизијску камеру. Главна разлика између видео камере и термовизијске камере је опсег електромагнетног зрачења које снимају. Термовизијска камера снима инфрацрвени опсег зрачења који са собом носи информацију о температури објекта који га одашиља. Термовизијске камере нашле су примену у системима за препознавање и заштиту пешака. Очита предност термовизијских камера је неосетљивост на боје и ефикасност ноћу. Мана им је примена у условима када је температура средине слична температури објеката који се препознају, рецимо током топлог дана. Слика 4 приказује пример примене термовизијске камере за препознавање пешака ноћу. На слици са десне стране се јасно може уочити пешак поред пута док се на слици са леве стране, услед недостатка светлости, једва наслућује.

Слика 4: Поређење снимка видео камере и термовизијске камере ноћу

2.3.Примери подсистема за напредну помоћ возачу

Подсистеми за напредну помоћ возачу се најчешће ослањају на комбинацију неколико већ познатих и проверених алгоритама. Уколико су у питању подсистеми који обрађују видео, то је углавном издвајање ивица праћено са препознавањем жељених облика. Системи за упозорења најчешће прате сигнал директно са сензора и траже образац у њему помоћу неког од класификатора. Неретко је успешност обрадâ побољшана примењивањем знања из области IoT-а (енгл. *Internet of Thing*) и машинског учења. Научна област која се бави изучавањем величина које би могле носити информацију о опасности у спољашњој средини, како их мерити и анализирати, веома је актуелна и доприноси у овој области се тек очекују. У наставку следи опис неколико карактеристичних подсистема чија је улога помоћ возачу.

Када започињемо причу о системима за помоћ возачу, не би било фер да не почнемо са ABS-ом (од енгл. *Anti-lock Braking System*). Овај систем има задатак да спречи блокирање точкова приликом кочења што се углавном дешава приликом оштрог кочења или кочења на клизавој подлози. Употребом ABS-а постиже се далеко боља управљивост током кочења. ABS константно прати угаону брзину точкова и у случају наглог пада препознаје да је дошло до проклизавања те отпушта кочницу на кратки временски период. Данас, ABS је део стандардне опреме свих нових аутомобила. [4]

Системи за електронску контролу проклизавања возила (енгл. *Electronic Stability Program*), познати под акронимом ESP. Имају улогу у спречавању заношења возила. У случају када путања возила не прати путању која је задата углом управљања волана, ESP ће реаговати примењивањем кочнице на одговарајућем точку како би пробао да предупреди заношење. ESP мери брзину окретања сваког точка, жељени смер кретања на основу угла волана и заокретање аутомобила око усправне, попречне и уздужне осе, па упоређивањем добијених вредности доноси одлуку о циљаном кочењу појединих точкова. Напредни ESP системи имају могућност и контроле над гасом, тј. мотором, па могу смањити обртни моменат, уколико ће то резултовати повећањем вуче. [5]

Систем за аутоматско прилагођавање крстареће брзине (енгл. *adaptive cruise control*) представља надоградњу познатих система за одржавање крстареће брзине (енгл. *cruise control*). Циљ овог система је одржавање безбедне удаљености од возила испред прилагођавањем тренутне брзине. У ту сврху користе се искључиво сензори постављени на возило којим се управља, дакле не остварује се комуникација са спољашњим уређајима. Најчешће, системи за аутоматско прилагођавање брзине користе LIDAR сензор. па на основу измерене удаљености смањују тренутну брзину или је одржавају на задатом нивоу. Овакви системи погодни су за вожњу на ауто-путевима и умањују вероватноћу чеоног судара. Напреднији облици ових система имају и могућност примене у „стани-крени“ вожњи у гужви. [6]

Системи за препознавање поспаности возача имају задатак да прате будност возача и закључују о његовој прибраности. Постоји више начина препознавања поспаности возача. Један од најуспешнијих алгоритама прати угао окретања волана и у том сигналу препознаје образац који је карактеристичан за поспане људе. Други алгоритми ослањају се на снимак са камере која снима возача и прати карактеристичне тачке на његовом лицу или препознаје положај очних капака. У случају препознавања поспаног возача систем реагује дрмусањем волана или звучним сигналом. Овакви системи могу бити јако корисни при дугим вожњама. [7]

3. Обрада видеа употребом *OpenCV* рачунарске библиотеке

OpenCV представља рачунарску библиотеку намењену дигиталној обради слике, машинском учењу и рачунарској визији. Оптимизована је за извршавање на системима у реалном времену и лако се укључује у апликације писане у *C++, python* и *Java* програмским језицима. Њен изворни код је јаван (енгл. *open source*) и објављена је под BSD лиценцом (енгл *Berkeley Software Distribution license*), што значи да се слободно може користити и у академске и у комерцијалне сврхе. *OpenCV* рачунарска библиотека подржана је на *Windows, Linux, Android* и *Mac OS* оперативним системима. [8]

OpenCV рачунарска библиотека је већ пронашла своју улогу у занимљивим решењима водећих компанија, попут спајања фотографија усликаних возилима *Google Street View*-а. У имплементацији овог рада, *OpenCV* ће бити главно оруђе за обраду слике.

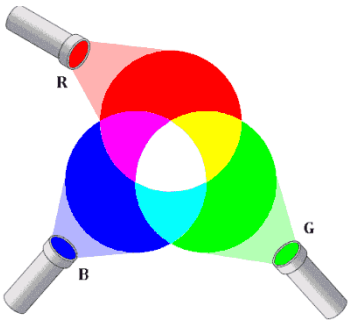
3.1. Представљање слика

Људско око је један од најосетљивијих и најсложенијих органа у људском организму. Око нам омогућава да разноврстан садржај електромагнетног зрачења из спољашње средине доживимо као јасну слику. Електромагнетни таласи различитих таласних дужина у људском оку побуђују хемијске реакције. Овакве реакције резултују стварањем акционог потенцијала који путем осећајног нерва путује од ока до коре великог мозга, где се ствара визуелна интерпретација, слика. Како рачунари баратају једино цифрама, оваква интерпретација сликâ им није позната. Да бисмо омогућили складиштење, обраду и приказивање слика на рачунару, нужна је вредносна и просторна квантизација.

3.1.1. Вредносна квантизација и системи боја

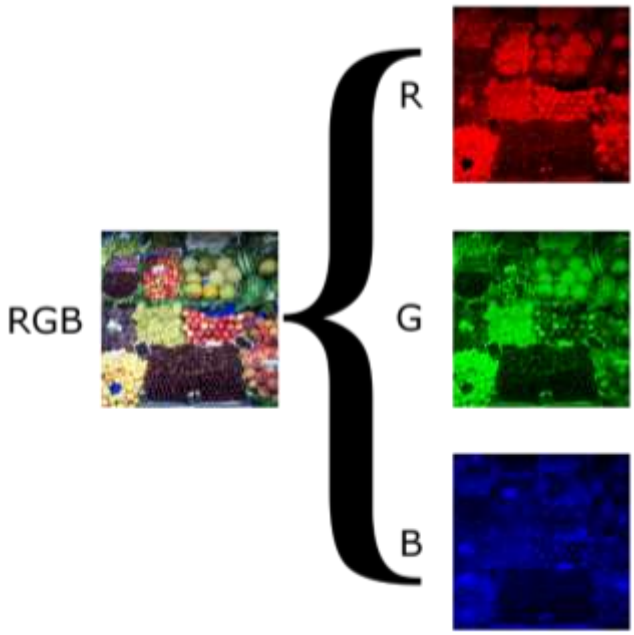
Вредносна квантизација подразумева додељивање јединственог бинарног кода свакој од боја. Од дужине кода зависи колико различитих боја је могуће представити тим типом кодовања. Број битова којим се кодира свака од боја назива се дубина боје (енгл. *color depth*). Скуп правила по којима се свакој боји додељује бинарни код назива се систем боја (енгл. *color space*). По правилу, у истом систему боја и истих димензија, слике веће дубине доживљавамо као квалитетније. Најчешћи системи боја су:

- RGB (скраћено од енгл. *Red Green Blue*) систем боја је најчешћи систем боја. Свака боја, у RGB систему, је представљена кроз три компоненте: црвену, зелену и плаву. Ове компоненте су независне једна од друге и узимају реалне вредности из опсега од 0 до 1. Како смо поменули да се боје кодирају бинарним вредностима, овај опсег се линеарно пресликава на скуп вредности од 0 до 2n, где је n дубина боје слике. Захваљујући чињеници да је RGB систем боја адитиван, у стварности сваку од боја је могуће приказати користећи три светлеће диоде које емитују црвену, зелену и плаву светлост. Адитивност боја у RGB систему илустрована је на слици 5.



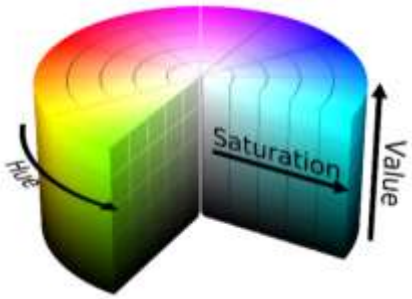
Слика 5: Илустрација адитивности RGB система боја

Матрице које садрже вредности кодова сваке од компоненти називамо равнима боја. Свака од равни боја оваквог система не носи довољно информација о целокупној слици и често није могуће препознати садржај слике само на основу једне равни. Слика 6 приказује пример слику богату разноврсним бојама и садржај свих њених RGB равни. *OpenCV* рачунарска библиотека складишти RGB слике у нешто измењеном редоследу. *OpenCV* рачунарска библиотека користи BGR систем боја.



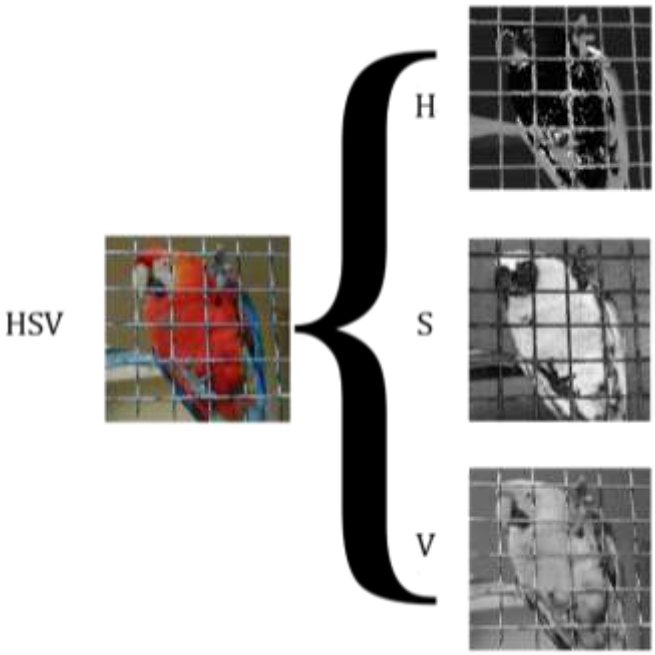
Слика 6: Слика представљена у RGB систему боја и њене равни

- HSV (скраћено од енгл. *Hue Saturation Value*) систем боја у којем је код сваке боје подељен у три сегмента: нијанса (енгл. *hue*), засићење (енгл. *saturation*) и осветљење (енгл. *value*). Овакав метод груписања компоненти боја близак је начину на који човек доживљава боје. Његова главна предност јесте чињеница да је информација о нијанси одвојена од информације о осветљености боје. Опсег вредности за нијансу је од 0 до 360 и ова вредност дефинише боју онако како је људски мозак доживљава. Засићење се представља реалном вредношћу у опсегу од 0 до 1 и оно носи информацију о уделу сиве боје. Осветљење представља интензитет боје и узима вредност из скупа бројева од 0 до 100 где 0 представља потпуно црну боју независно од вредности осталих компоненти. [9] Организацију HSV система боја је најлакше визуелизовати када се боје распореде у цилиндричном координатном систему као на слици 7.



Слика 7: Распоред боја у HSV систему боја

Овако постављен систем боја погодан је за рачунарску сегментацију слике када је од интереса издвајање области одређене боје. На слици 8 се види како области исте боје имају сличне вредности нијансе. У имплементацији овог рада, HSV систем боја ће бити коришћен за потребе издвајања белих и жутих области коловозних трака.



Слика 8: Слика представљена у HSV систему боја и њене равни

- Систем боја у нивоима сиве (енгл. *grayscale*) је систем боја у којем се бинарни код сваке боје одређује на основу осветљености те боје. Овакав систем боја садржи само једну компоненту коју називамо интензитет. Интензитет се представља реалном вредношћу у опсегу од 0 до 1 где 0 представља црну боју, а 1 представља белу боју. Вредност интензитета се линеарно пресликава у бинарне вредности од 0 до 2^n где је n дубина боје слике. Слика 9 приказује изглед слике у боји и изглед слике након трансформације у систем боја у нивоима сиве.

Слика 9: Изглед слике у боји и након трансформације
у систем боја у нивоима сиве

Слике представљене у систему боја у нивоима сиве погодне су за даљу обраду па често представљају резултат фазе предпроцесирања и припреме за примену алгоритама попут препознавања ивица.

- Црно-бели систем боја је најчешћи бинарни систем боја. У овом систему боја дефинисане су само две боје, црна и бела. Како су дефинисане само две боје, дужина кода којим обе боје могу бити једнозначно одређене је један. Слика 10 приказује изглед слике у боји и у црно-белом систему боја.

Слика 10: Изглед слике у боји и у црно-белом систему боја

Црно-беле слике се брзо обрађују па су често резултат сегментације слика представљених у другим системима боја. Сегментацијом се из почетне слике издвајају области које задовољавају постављени услов.

Једну слику могуће је представити и запамтити у сваком од система боја. Познате су и функције које описују пресликавање кодова из једног у други систем боја. За потребе пребацивања слике из једног у други систем боја, *OpenCV* рачунарска библиотека нуди функцију *cvtColor*. Ова функција прима три аргумента, изворну слику, слику у коју се смешта резултат и трећи аргумент који означава из којег у који систем боја се врши конверзија.

3.1.2. Просторна квантизација

Просторна квантизација остварена је дељењем слике на велики број једнобојних региона. Такви региони називају се пиксели (енгл. *pixels*) и сваком од пискела додељена је једна квантована вредност боје. Уређени пар који чине број пиксела који стају у ширину и број пиксела који стају у висину слике назива се резолуција. По правилу, у истом систему боја са истом дубином, слике веће резолуције доживљавамо као квалитетније. Слика 11 приказује утицај резолуције на квалитет слике.

Слика 11: Утицај резолуције на квалитет слике

Када сумирамо вредносну и просторну квантизацију, јасно је да је слика, у рачунару, представљена у облику матрице. Резолуција слике диктира димензије матрице, а сваки елемент матрице садржи бинарни код боје једног пиксела.

За потребе складиштења сликâ, у *OpenCV* рачунарској библиотеци, дефинисана је *Mat* класа. Ова класа доступна је од верзије 2.0 поменуте библиотеке. *Mat* класа складишти два типа података, метаподатке о слици и показивач на матрицу у којој су смештене вредности пиксела. Матрица вредности пиксела често је дељена између више *Mat* објеката и копирање објеката ове класе имплицитно подразумева дељење матрице вредности између оригинала и копије. На овај начин избегава се заузимање новог меморијског простора при прослеђивању слике као аргумента функције. Како обрада слике најчешће подразумева позивање низа смислених функција над једном сликом, на овај начин се штеди драгоцено време. [10]

3.2. Канијев детектор ивица

Канијев детектор ивица (енгл. *Canny edge detector*) представља добро познати рачунарски алгоритам за издвајање ивица на слици. Развио га је аустралијски научник Џон Кани (енгл. *John Canny*). Алгоритам је развијен 1986. године за време Канијевог рада на MIT-у (скраћено од енгл. *Massachusetts Institute of Technology*). Издвајање ивица овим алгоритмом се показало успешнијим од издвајања ивица на основу градијента.

Канијев детектор ивица осмишљен је са циљем да се:

- Максимално умањи грешка издвајања ивица
- Тачке које чине једну ивицу уско групишу дуж средишта те ивице
- За јединичну ивицу добије само један одзив

Након истраживања, Кани је дошао да закључка да су горе наведени услови испуњени када се обрада дате слике проведе кроз низ узастопних корака. Пре описа засебних корака важно је напоменути да Канијев алгоритам очекује почетну слику која је представљена у систему боја у нивоима сиве.

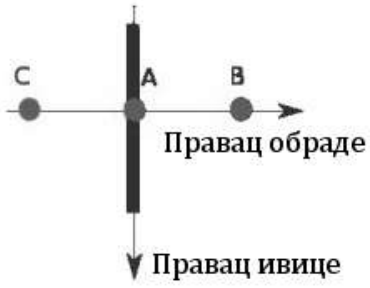
Први корак у примени Канијевог алгоритма јесте отклањање шума. Познато је да су сви алгоритми за детекцију ивица веома осетљиви на шум. Нарочите проблеме може направити импулсни шум. Канијев алгоритам користи Гаусов филтер за отклањање шума. Ширина Гаусовог филтера зависи од резолуције слике. Повећање ширине Гаусовог филтра умањује осетљивост алгоритма на шум по цену губитка тањих и краћих ивица на слици.

Други корак Канијевог алгоритма за издвајање ивица подразумева примену Собеловог оператора. Практично ово је корак у којем се од почетне слике добија слика која издваја ивице. У овом кораку се, на слику која је добијена након примене првог корака, независно примењују Собелов оператор за издвајање водоравних ивица и Собелов оператор за извдвајање усправних ивица. Две слике које се добију као резултати филтрирања се затим стапају у једну коришћењем формуле $G = \sqrt{G_x^2 + G_y^2}$, где је G_x слика која садржи водоравне ивице, а G_y слика која садржи усправне ивице. Овако добијена слика се користи као улазна слика за наредни корак Канијевог алгоритма. Слика 12 илуструје почетну слику, међурезултате и слику која представља поменути збир. Додатно, резултат овог корака је још једна матрица која ће бити од користи у наредном кораку обраде. Ова матрица носи информацију о правцу пружања издвојених ивица и добија се по формули $theta = \arctan(G_y/G_x)$.

Слика 12: Примена Собеловог оператора као један од корака Канијевог алгоритма

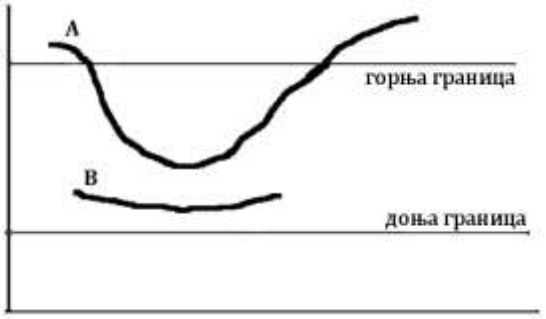
Наредни кораци имају за циљ да још јасније издвоје ивице које задовољавају набројане критеријуме. Трећи корак у целокупној обради подразумева потискивање локалних немаксимума са слике која је добијена као резултат претходног корака. Потискивање локалних немаксимума узрокује истањавање ивица.

За обраду у овом кораку користимо матрицу *theta* која је била споредни резултат претходног корака. Вредности у овој матрици носе информацију о орјентацији ивице. Пре употребе матрице врши се редукција информација тако што се све вредности квантизују у четири правца, водоравни, усправни и два правца по дијагонали. Пошто нам је позната орјентација ивице можемо да потражимо локалне максимуме, а вредности које нису локални максимуми заменимо нулама. Пролазимо кроз целу слику, пиксел по пиксел и испитујемо два од осам првих суседа сваког пиксела. Која два пиксела ћемо посматрати зависи од вредности у матици *theta* односно од правца којим се простире ивица. Посматрамо пикселе који су нормални на правац простирања ивице. На пример, уколико је ивица усправна, посматрамо први леви и први десни пиксел пискела који обрађујемо. Уколико је вредност пиксела који се обрађује већа од вредности оба суседна пиксела, пиксел задржава своју вредност, иначе добија вредност нула. Оваквим алгоритмом задржавамо само локалане максимуме у правцу пружања ивице. Слика 13 илуструје описани поскупак. Пиксел А задржава своју вредност само уколико је њихова вредност већа и од вредности пиксела С и од вредности пиксела В.



Слика 13: Потискивање локалног немаксимума

Последњи корак има за задатак крајње издвајање. Излаз овог корака је бинарна слика која је уједно и резултат примене Канијевог детектора ивица. За примену овог корака потребне су нам две граничне вредности које одређују горњу и доњу граничну вредност сегемнтације. Сви пиксели чија је вредност изнад горње границе препознајемо као пикселе који сачињавају ивице и одмах им додељујемо белу боју. Слично, све пискеле чија је вредност испод доње границе одбацујемо и додељујемо им црну боју. Питање је шта са пикселима чија је вредност између доње и горње границе. Ове пискеле препознајемо као делове ивица уколико у свом суседству имају барем један пиксел који је класификован као део ивице. Овакав поступак одлучивања се ланчано наставља. Слика 14 осликава описано закључивање. Ивицу А препознајемо као ивицу док ивицу В одбацијемо, јер нема повезаних пиксела чија је вредност већа од горње границе. Овакав поступак се још назива и сегментација са хистерезисом. [11]



Слика 14: Издвајање ивица са хистерезисом на основу горње и доње граничне врендости

У оквиру рачунарске библиотеке *OpenCV* доступна је функција која спроводи описану обраду и издвајање ивица. Поменута функција се назива *Canny* и прима 5 аргумената. Прва два аргумента намењена су улазној и излазној слици, дакле, слика која је резултат обраде враћа се кроз прослеђени аргумент. Трећи параметар одређује ширину Гаусовог филтра који ће бити примењен за отклањање шума и замућивање улазне слике. Последња два аргумента служе за прослеђивање горње и доње границе за потребе сегемнтације са хистерезисом. Вредности граница битно утичу на исход примене Канијевог алгоритма. Препорука је да однос вредности ова два параметра буде између 2 и 3. Слика 15 приказује утицај промене вредности параметра на крајњу сегментацију. Крајње очекивано, са повећањем вредности граница остају само значајно изражене ивице. [12]

Вредности поменутих параметара обраде, у току израде овог рада, одређивани су експериментално.

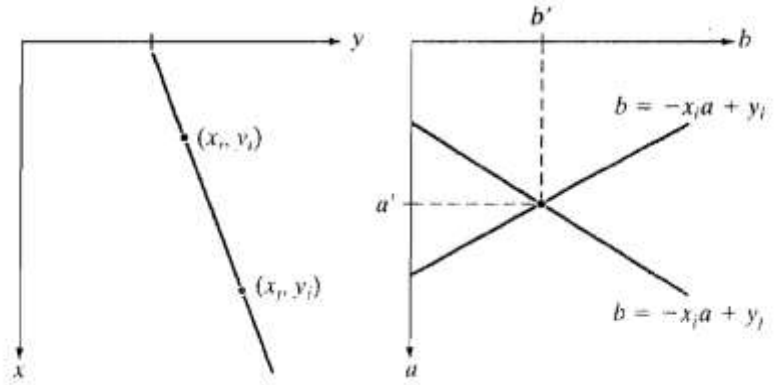
Слика 15: Утицај параметара Канијевог алгорима на успешност издвајања ивица

3.3. Хафова трансформација

Хафова трансформација представља технику за издвајање облика. Назива се трансформацијом зато што пресликава просторни домен у параметарски и у параметарском домену спроводи препознавање вредности параметара који одговарају облицима у просторном домену. Хафова тренсформација названа је по америчком истраживачу Полу Хафу (енгл. *Paul Hough*) који је ову трансформацију описао још 1962. године. Након њега, трансформацију су 1972. године додатно истражили и унапредили амерички научници Ричард Дуда (енгл. *Richard Duda*) и Питер Харт (енгл. *Peter Hart*). Хафова трансформација је постала доста популарнија пошто је своју примену нашла у алгоритмима за препознавање и издвајање облика у компјутерској визији.

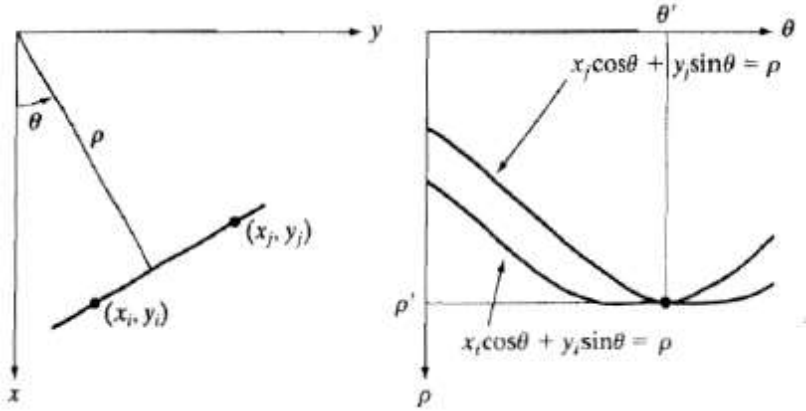
Хафову трансформацију ћемо објаснити на примеру препознавања правих линија на слици. Управо је ово и сценарио у којем ће Хафова трансформација бити употребљена током имплементације решења овог рада.

Посматрајмо праву у просторном домену, у Декартовом координатном систему. Једну праву чине све тачке чије координате задовољавају једначину праве: $y = ax + b$. Уређени пар a и b једнозначно одређује праву. Уколико бисмо, уместо у Декартовом координатном систему, праву представили у координатном систему параметара a и b, права би била графички представљена као тачка. Могуће је поставити и инверзну једначину: $b = -xa + y$ на основу које можемо закључити да се тачка (x, y) из просторног домена пресликава у праву у параметарском домену. Знајући да је кроз једну тачку могуће провући бесконачно права, овакав закључак је очекиван Све праве у параметарском домену које представљају тачке једне праве у просторном домену се у параметарском домену секу у једној тачки. Пресечна тачка налази се на координатама a и b. Слика 16 илуструје описану идеју. Са слике 16 се лако закључује да се претрага за правом у просторном домену своди на претрагу за пресечним тачкама у параметарсом домену.



Слика 16: Пресликавање праве из просторног у параметарски домен

Недостатак пресликавања описаног у претходном пасусу јесу усправне праве чији параметар a, коефицијент правца, има велике вредности. Овај недостатак је могуће избећи уколико праву представимо у другачијем облику: $xcos\theta + ysin\theta = \rho$. Графичка представа параметара θ и ρ приказана је на слици 17. Поново се права из просторног домена пресликава у тачку у параметарском домену. Међутим, по претходној формули закључујемо да се сада тачка из просторног домена пресликава у синусоиду у параметарском домену (слика 17).



Слика 17: Пресликавање праве из просторног у параметарски домен

Да бисмо применили горе описану идеју, формирамо матрицу која ће представљати $\theta\rho$ параметарски простор. Вредности за угао θ су у опсегу од -90 до 90. Вредности за растојање ρ се најчешће узимају од $-\sqrt{2}D$ до $\sqrt{2}D$ где је D дужина дијагонале у пикселима. Са овако постављеном матрицом, пролазимо кроз слику и попуњавамо је. Потребно је напоменути да се Хафова трансформација примењује само на бинарним сликама. Улазна слика за примену Хафове трансформације је најчешће добијена применом неког од алгоритама за издвајање ивица. За сваки пиксел чија вредност није нула, на основу његове позиције на слици, прерачунамо вредности θ и ρ параметара и унесемо синусоиду у резултујућу матрицу. Након проласка кроз целу слику, тачке нагомилавања у резултујућој матрици представљају комбинације θ и ρ параметара који одговарају линијама на почетној слици. На слици 18 приказана је слика на којој се примењује Хафова трансформација и резултујућа слика. Јасно се види да се на резултујућој слици издваја осам тачака нагомилавања које одговарају линијама на почетној слици. Може се и приметити да дужим усправним линијама одговарају светлије тачке пресека на резултујућој слици. [13]

Слика 18: Илустрација примене Хафове трансформације

За потребе примене Хафове трансформације, рачунарска библиотека *OpenCV* нуди две функције, *HoughLines* и *HoughLinesP*. *HoughLinesP* је нешто ефикаснија и због тога је у овом раду избор пао на њу. Ова функција прима седам аргумената. Као и код већине функција, први аргумент резервисан је за слику која се обрађује, а други за повратну вредност. Повратна вредност, у овом случају, је низ уређених четворки (x_1, y_1, x_2, y_2), где су (x_1,y_1) и (x_2, y_2) координате почетка и краја сваке од препознатих линија. Као трећи и четврти аргумент, горе поменута функција, прима вредности које одређују која ће бити резолуција за параметре θ и ρ. Пети аргумент дефинише коју минималну акумулирану вредност препознајемо као линију у просторном домену. На крају, шести и седми аргументи додају ограничења у виду минималне дужине коју линија мора остварити да би била прихваћена и максималног прекида линије које може да се толерише. [14]

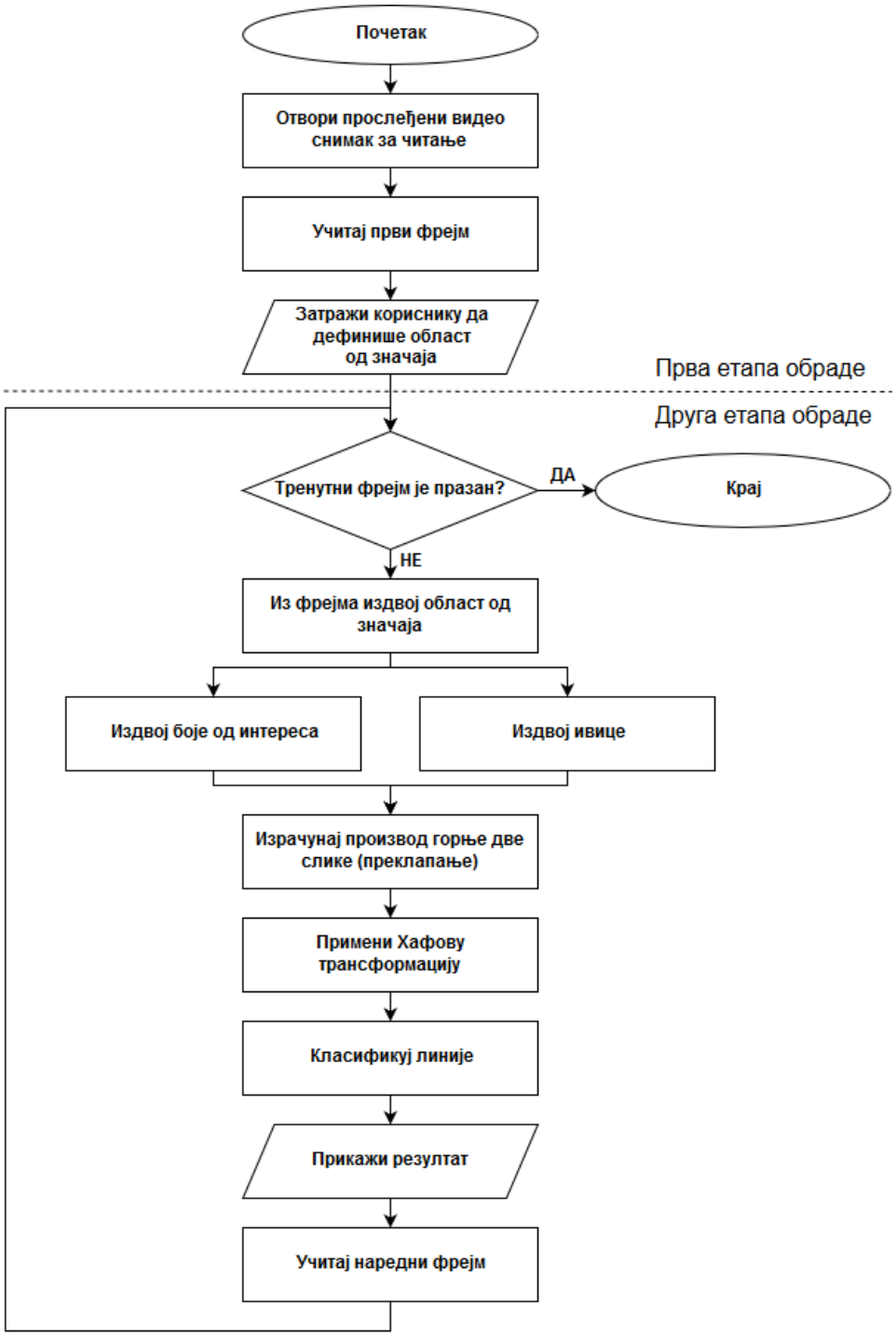
4. Предлог решења

Циљ овог рада је развој алгоритма за препознавање разделних линија коловозних трака на основу видео снимка који се добија са камере постављене на ветробранско стакло аутомобила. Развијени алгоритам комбинује знања из области дигиталне обраде слике, препознавања облика и рачунарске технике. Програм, који примењује предложени алгоритам, написан је у C++ програмском језику, а за потребе дигиталне обраде слике користи позиве из рачунарске библиотеке *OpenCV.*

Развијени алгоритам се примењује на сваки од фрејмова (енгл. *frame*) и исход обраде су две линије које представљају леву и десну разделну линију. Слика 19 приказује резултат обраде једног од фрејмова и издвојене разделне линије. Уколико није могуће препознати неку од разделних линија, било да је нема или је тешко уочљива, програм не треба ништа да прикаже на том месту и да пређе на обраду наредног фрејма. Обрада тренутног фрејма мора да се заврши пре пристизања наредног.

Слика 19: Резултат обраде једног фрејма

Програм се одвија кроз две етапе. Прва етапа, која се извршава само једном, служи за подешавања параметара за примену алгоритма и прилагођавање програма положају камере. Друга етапа је главни део програма у којем се, у петљи, примењује развијена обрада и препознавање. У другој етапи, алгоритам се кружно примењује на сваки од фрејмова до тренутка завршетка видео снимка. Дијаграм тока програма приказан је на слици 20.



Слика 20: Дијаграм тока програма

Први део тока програма односи се на одабир области од значаја. Област од значаја, позната и као ROI (од енгл. *Region Of Interest*), део је слике у којој очекујемо да ће се тражени облик наћи, у овом случају разделне линије коловозних трака. Након дефинисаног ROI-а програм обрађује само део слике који је издвојен. На овај начин, брзина обраде је знатно већа јер су димензије подслике мање. Још важније, успешност проналажења је далеко већа, јер увођењем области од значаја занемарујемо детаље ван те области који би могли да наведу програм на погрешан закључак.

Одабир области од значаја уведен је како би програм могао успешно да обрађује видео материјал са више различитих камера. ROI зависи од позиције и оријентације камере на ветробранском стаклу. У стварним ситуацијама, камера се, за време производње, поставља у аутомобил и њена позиција је позната па је и област од интереса предефинисана. Погрешним одабиром области од значаја могу се добити лошији резултати. Ова област бира се тако да њена ширина буде нешто већа од ширине коловозне траке, а њена дубина треба да обухвати око 20m коловоза испред аутомобила.

На слици 21 приказана су три примера одабира области од значаја. Пример на левој слици је лош зато што је висина области превелика па обухвата и детаље који нису од значаја. Пример на десној слици је лош зато што област од значаја не обухвата читаву ширину коловозне траке и лева разделна линија излази из области. Пример на слици у средини приказује добар избор области од значаја.

Слика 21: Одабир области од значаја, одабири на сликама лево и десно су лоши

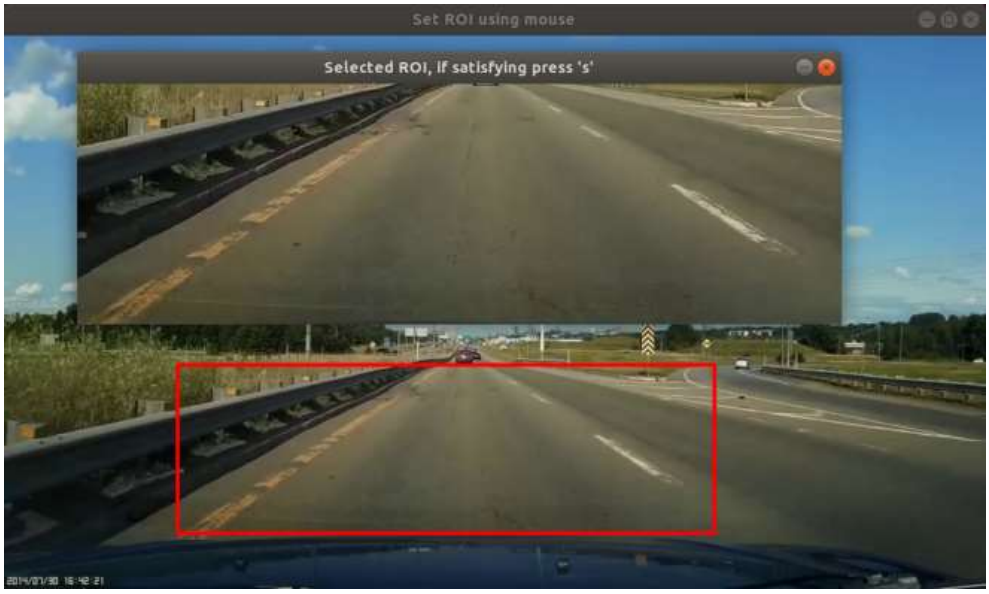
Након дефинисане области од значаја, програм ће наставити ток у петљи. У свакој итерацији петље примењиваће се низ корака који треба да издвоји разделне линије. У једној итерацији обрађиваће се један фрејм са видео снимка. На крају итерације кориснику се приказује резултати и прелази се на обраду наредног фрејма. Главна петља се понавља све док има фрејмова који се учитавају из видео тока.

5. Имплементација решења

5.1. Одабир области од значаја

Одабир области од значаја уведен је како би корисник ручно подесио програм за обрађивање прослеђеног снимка. Област се подешава тако што се кориснику прикаже изглед првог фрејма и чека се на његову реакцију. Корисник треба да, користећи миша, дефинише који део слике је од значаја. Област се дефинише притиском дугмета миша и превлачењем показивача до места отпуштања дугмета.

У сврху употребе миша, дефинисан је ослушкивач догађаја миша (енгл. *mouse listener*). Овај ослушкивач памти координату на којој је корисник кликнуо и координату на којој је корисник пустио дугме миша. Са ова два податка је област од интереса једнозначно одрађена. Уколико је област исправно дефинисана, тј. уколико границе области нису изван слике, кориснику се приказује дефинисана област и чека његова потврда. Уколико област није исправно дефинисана, захтева се поновно означавање. Слика 22 приказује изглед екрана за време одабира ROI-а.



Слика 22: Изглед екрана за време одабира области од значаја

Програм је блокиран све док корисник исправно не означи област од значаја и не притисне дугме „S“ на тастатури, чиме се дефинишу границе области од значаја и оне се памте у меморији. Информације о границама области од значаја се не мењају до краја тока програма. Запамћене границе области од значаја примењују се на сваки наредни фрејм.

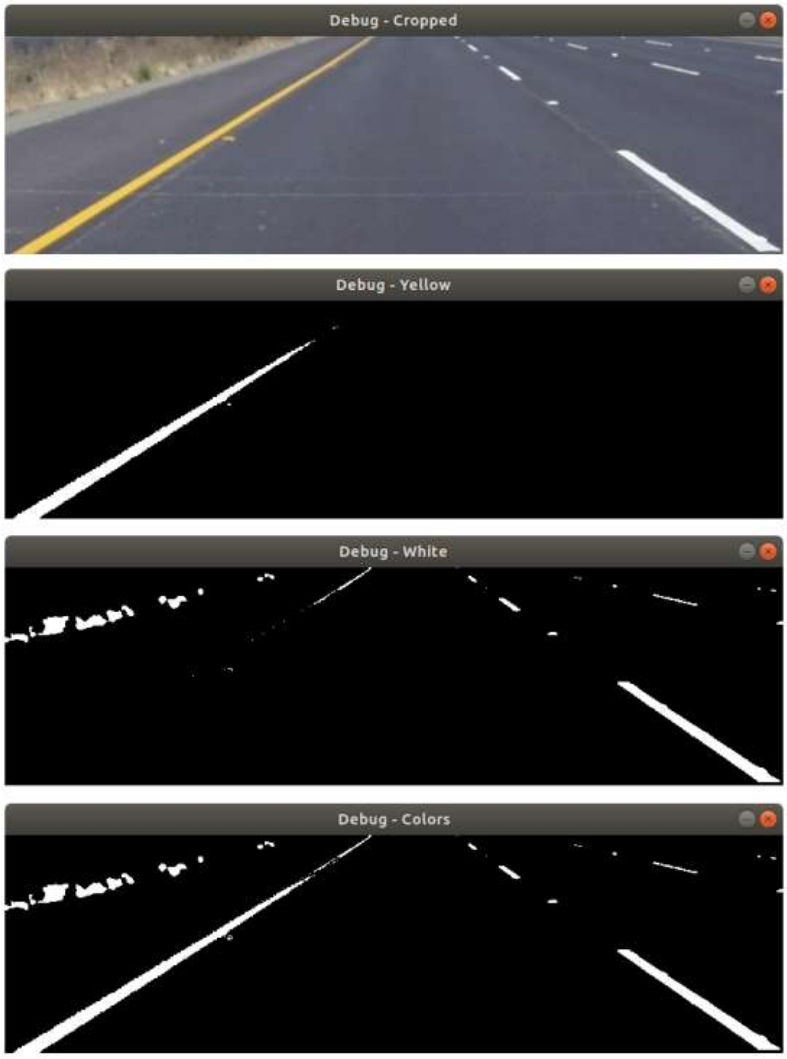
5.2. Издвајање боја

Услед превеликог броја могућих детаља на слици није довољно само издвојити ивице, већ је нужно унети још неки ниво „заштите“. Пошто је познато да су разделне линије коловозних трака или жуте или беле боје, можемо се додатно заштити тако што ћемо тражити само ивице које се налазе у непосредној близини белих и жутих региона на слици. Из слике која се добија након исецања области од интереса, независно ћемо издвојити области беле и жуте боје, а затим сабрати те две слике.

За издвајање боја од велике помоћи је разумевање начина на који су слике представљене у меморији рачунара, о чему више пише у поглављу 3.1. Боје је најлакше издвојити уколико је слика запамћена у HSV систему боја. Имајући то у виду, први корак је пребацивање слике из BGR система боја у HSV систем боја коришћењем *cvtColor* позива из *OpenCV* рачунарске библиотеке.

На слику која је запамћена у HSV систему боја, потребно је применити механизам који ће издвојити области чија је боја унутар граница које ограничавају жељену боју. Приликом одабира нећемо бити строги него ћемо укључити и део боја које су сличне жељеним. Жуту боју одређују вредности нијансе у границама од 50 до 70, међутим, програм је написан тако да прихвати све у границама од 40 до 80. Што се засићења и осветљености тиче, прихватљиве вредности за жуту боју су све изнад 50% максимума. Белу боју карактеришу високе вредности осветљености и ниске вредности засићености, па ћемо је у тој области и тражити. Прихватамо све области чија боја има засићеност мању од 25% и осветљеност већу од 75%.

Приликом издвајања боја од користи је функција *OpenCV* рачунарске библиотеке, *inRange*. Ова функција враћа бинарну слику која је истих димензија као и прослеђена слика. На почетној слици сви пиксели чија је боја била унутар прослеђених граница замењени су белом бојом. Сви пиксели чија се боја није уклопила у прослеђене границе замењени су црном бојом. По издвајању и жуте и беле боје, резултујуће слике ћемо сабрати. Слика која представља поменути збир је један од важнијих међурезултат у процесу издвајања разделних линија. На слици 23 приказан је процес издвајања боја. На самом врху је почетна слика, затим међурезултати и, на дну, резултујућа збирна слика.



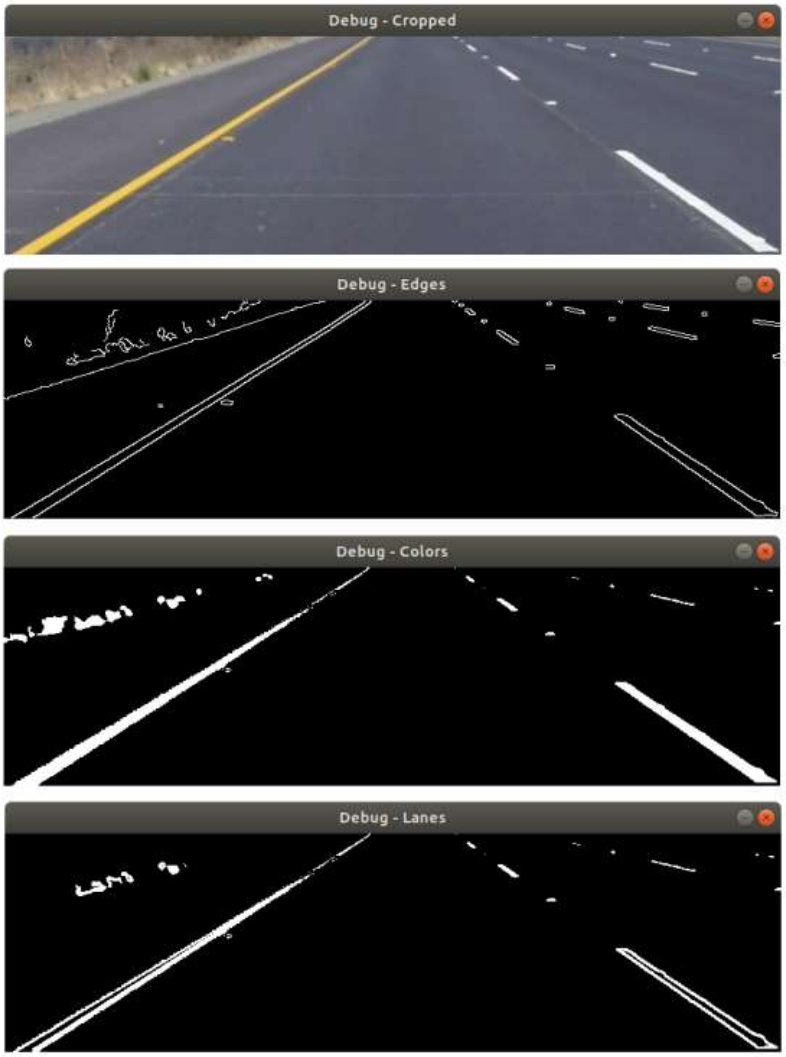
Слика 23: Почетна слика, међурезултати и резултат издвајања боја

5.3.Издвајање ивица

Паралелно са издвајањем боја одиграва се и издвајање ивица. Ивице ћемо издвајати са почетне слике, а не са слике која се добија након издвајања боја, јер ћемо на тај начин само поновити већ добијени резултат. Дакле, ивице издвајамо са слике која представља исечен тренутни фрејм. Како је издвајање ивица слично примени математичког извода, овај процес је јако осетљив на шум. Због тога ћемо улазну слику прво обрадити па онда применити Канијев детектор ивица. Желимо да елиминишемо све ситне детаље са слике, дакле, желимо да замутимо слику. У ту сврху користе се ниско пропусни филтри попут Гаусовог филта.

Након примене Гаусовог филта, којим је слика лишена ситних детаља попут пукотина, на путу, слику морамо додатно припремити пре примене Канијевог детектора ивица. Овај детектор се примењује на слике које су у систему боја у нивоима сиве па ћемо поново искористити *OpenCV* фунцкију *cvtColor*. Канијев детектор ивица сада лако издваја све скокове у осветљењу и враћа нам бинарну слику на којој су само издвојене ивице приказане белом бојом. О Канијевом детектору ивица је било речи у поглављу 3.2.

Слика која се добија као међурезултат издвајања ивица се затим множи са сликом која је међурезултат издвајања боја. Дакле, тражимо пресек ове две слике, односно све области у којима има ивица жуте или беле боје. На слици 24 приказана је почетна слика, испод ње, слика која се добија као међурезултат издвајања ивица и ниже слика која је међурезултат претходног корака. На дну је слика која представља поменути пресек.



Слика 24: Почетна слика, међурезултати и слика са извојеним разделним линијама

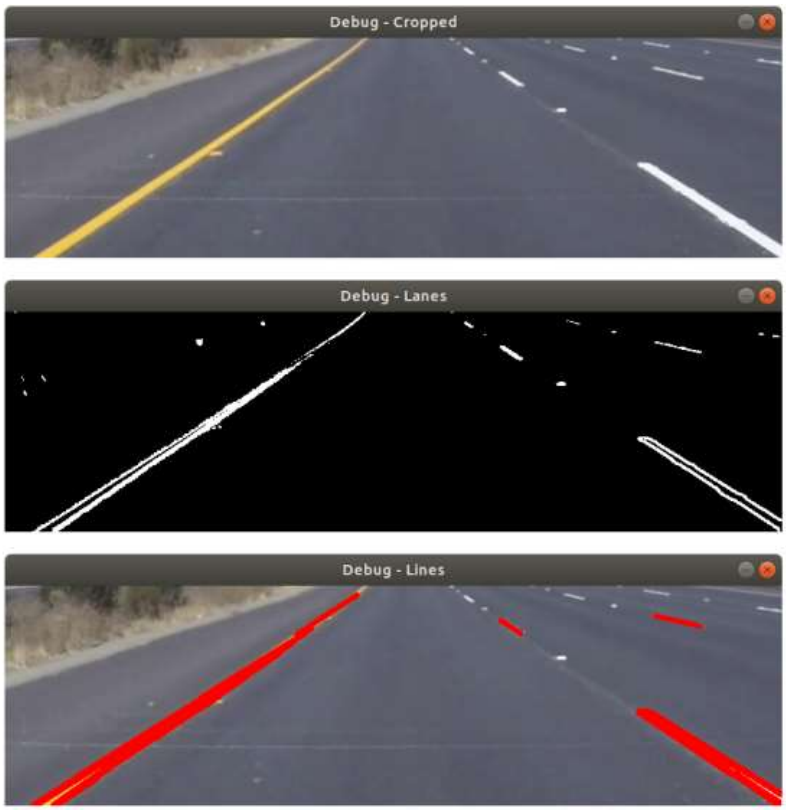
5.4. Примена Хафове трансформације

Након претходних корака обраде долазимо у завидну позицију у којој су са почетне слике уско издвојене области у којима можемо очекивати разделне линије. Међутим, још увек баратамо само визуелним приказима разделних линија. Потребно је да уведемо математички опис линија. Желимо да знамо једначине праве препознатих линија у координантном систему који је везан за камеру на ветробранском стаклу. Познавањем једначина правих, можемо да одредимо позицију аутомобила у саобраћајној траци. Ово су кључне информације на основу којих се аутомобил одржава у тренутној траци. За решавање оваквог задатка, идеална је Хафова трансформација.

Хафова трансформација се примењује на бинарним сликама и на основу њиховог садржаја одређује почетне и крајње тачке свих линија на слици. За спровођење Хафове трансформације користимо функцију *HoughLinesP*. Ова функција дефинисана је у *OpenCV* рачунарској библиотеци. Које критеријуме линије морају да задовоље да би биле препознате, зависи од параметара који су прослеђени при позиву. Више о Хафовој трансформацији описано је у поглављу 3.3.

Хафову трансформацију примењујемо на слику која је резултат пресека издвојених боја и издвојених ивица. Критеријуми за прихватање линија не смеју бити превише строги, јер ће у том случају бити прихваћене само „пуне“ линије, јер само оне имају довољну дужину. Додатно, уколико поставимо строге критеријуме, доводимо у питање препознавање линија које су даље од аутомобила и које се виде под оштријим углом. Постављамо критеријуме који ће свесно издвојити и неколико линија које уносе шум. Ове линије ћемо, ваљаном класификацијом, елиминисати у наредном кораку.

На слици 25 приказане су три слике. Као и обично, прва је почетна слика, тј. слика која је добијена издвајањем области од интереса са учитаног фрејма. Како се добија друга слика описано је у претходном кораку. Друга слика предстаља слику над којом се примењује Хафова трансформација. Трећа слика, на дну, приказује почетну слику са свим линијама које је препознала Хафова трансформација. На слици се јасно види да је једна разделна линија препозната као више краћих линија. Такође, можемо уочити да су издвојене и линије које припадају оближњим тракама. Те линије ће касније бити елиминисане класификацијом.

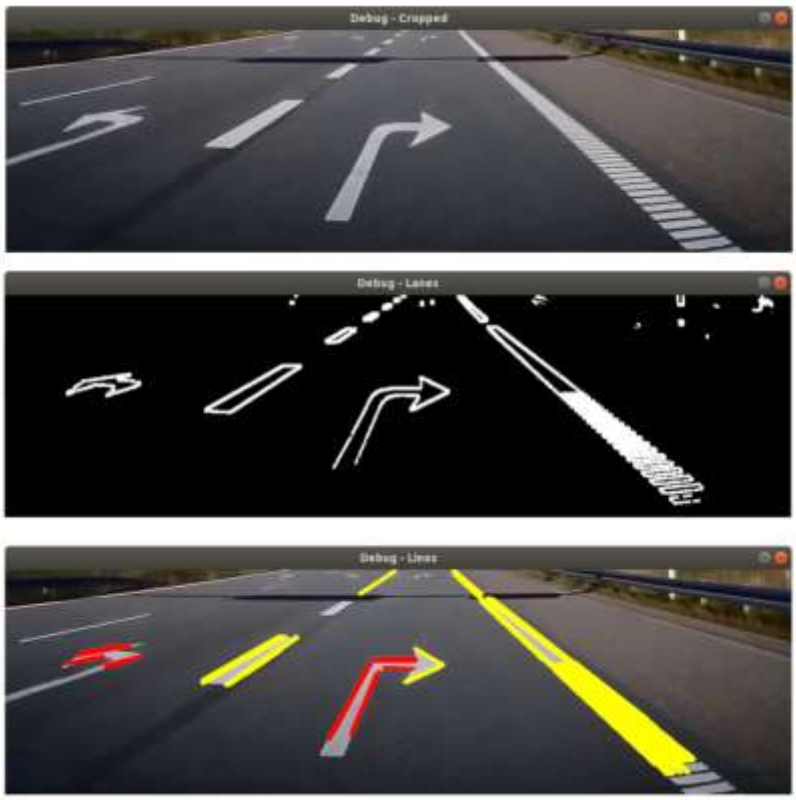


Слика 25: Резултат примене Хафове трансформације

5.5. Класификација линија

Када смо издвојили више линија које нам могу бити од интереса, потребно је да елиминишемо линије које представљају шум, а потом да јасно одредимо представнике леве и десне разделне линије. У ову сврху користимо бројчане вредности које су резултат Хафове трансформације. Подсећања ради, Хафова трансформација враћа низ уређених четворки које означавају координате почетних и крајњих тачака сваке линије.

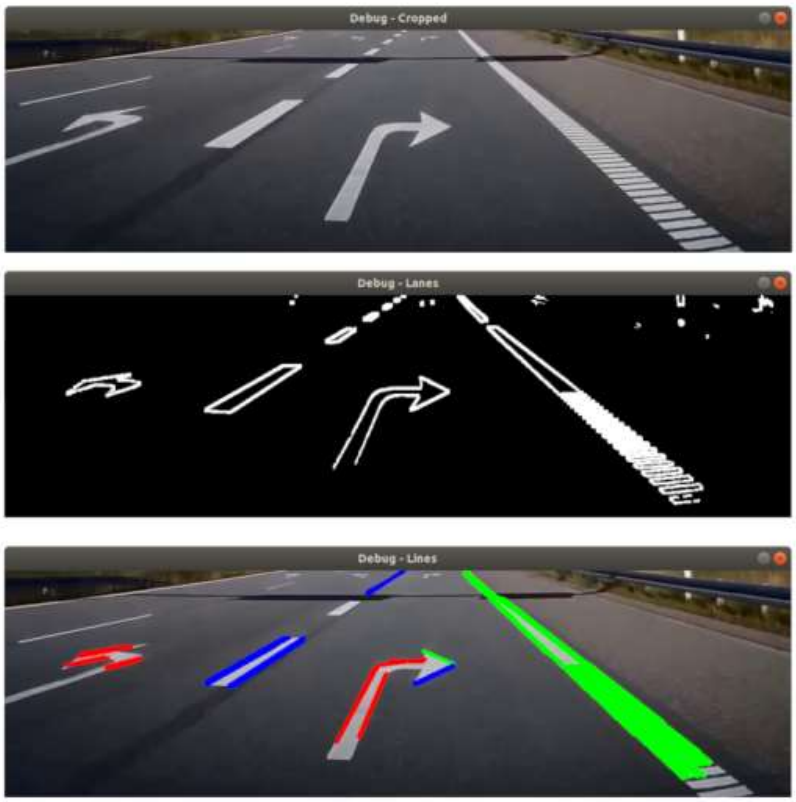
На основу координата почетних и крајњих тачака линија можемо израчунати коефицијенте правца сваке од линија. Први ниво одбране је одбацивање свих линија чији је коефицијент правца мањи од 0,36 или већи од 2,75 тј. све линије које су под нагибом мањим од 20° или већим од 70°. У овом кораку, класификација је строга. Главни циљ овог одбацивања је одбрана од детаља који се штампају на асфалту, али нису разделне линије, попут стрелица за престројавање. Слика 26 приказује задржане линије жутом бојом, а одбачене линије црвеноом бојом. Ова слика је одличан пример јер алгоритам не успева одмах да одстрани све шумовите линије. Те линије ће бити одбачене у наредним корацима класификације.



Слика 26: Одбацивање линија на основу нагиба

Пошто смо одбацили све линије чији се нагиб не уклапа у жељени образац, потребно је додатно да разврстамо линије. За почетак да одвојимо линије на два скупа, скуп левих линија и скуп десних линија. За ту сврху, поново ћемо искористити координате које је су одређене Хафовом трансформацијом. Овога пута, израчунаћемо у којој тачки би свака од линија пресекла X осу, односно доњу ивице издвојене слике. Све линије које би доњу ивицу пресекле лево од половине слике, сврстаћемо у леве, а све оне линије које би доњу ивицу пресекле десно од половине слике сместићемо у скуп десних линија. На овај начин не елиминишемо шум, још увек. Рецимо, са слике 26 је јасно да ће једна од ивица стрелице на путу бити сврстана у скуп левих линија, а друга у скуп десних линија, иако обе представљају шум. На слици 27 на наредној страници приказана је слика која се добија након разврставања линија. Линије које су одбачене у претходном кораку нису сврстане ни у једну групу.

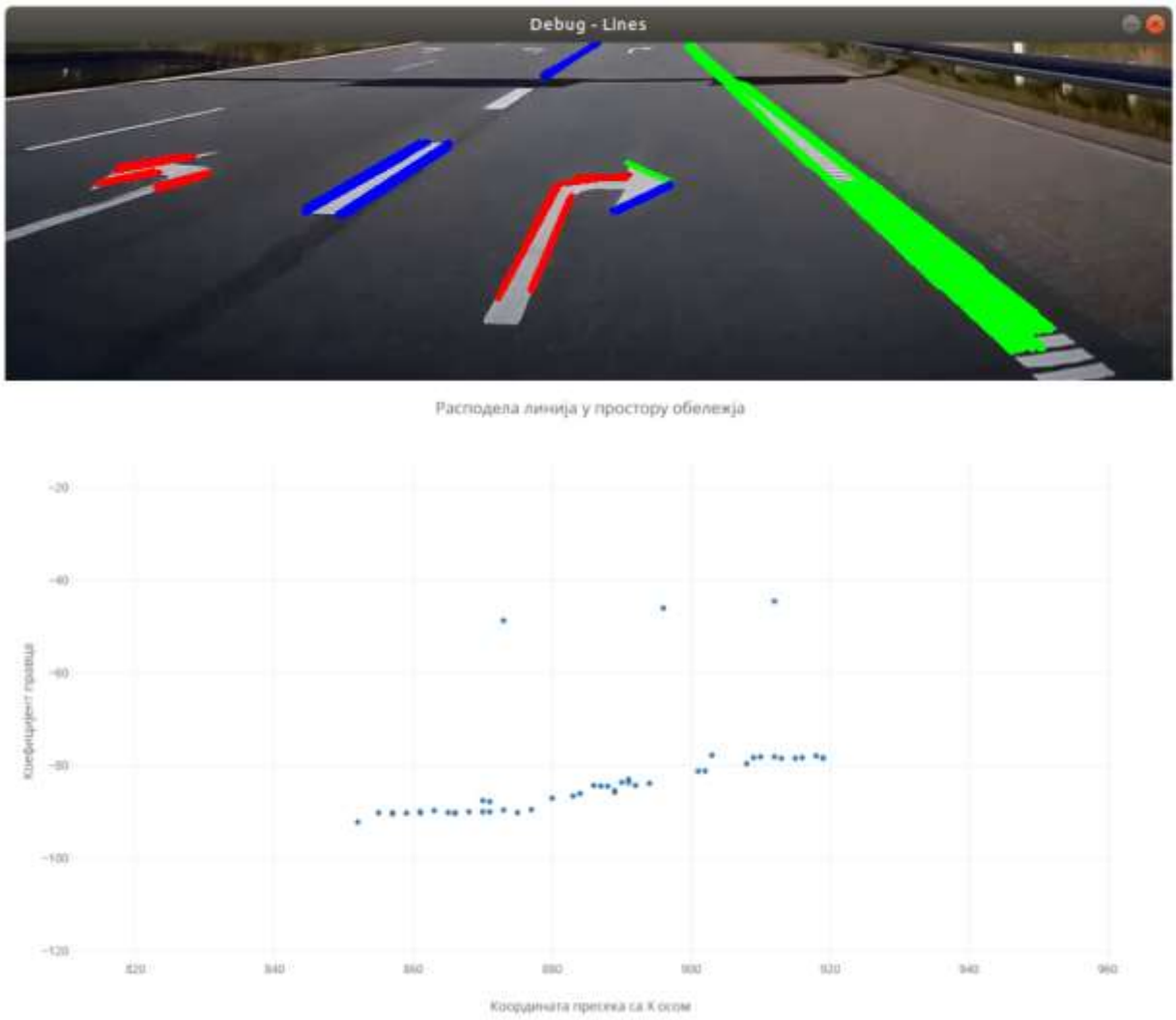
Податке о коефицијентима праваца и пресечним тачкама са X осом ћемо запамтити. Ова обележја ће бити од пресудног значаја у наредном кораку.



Слика 27: Линије подељене у скупове левих линија, десних линија и линија шума

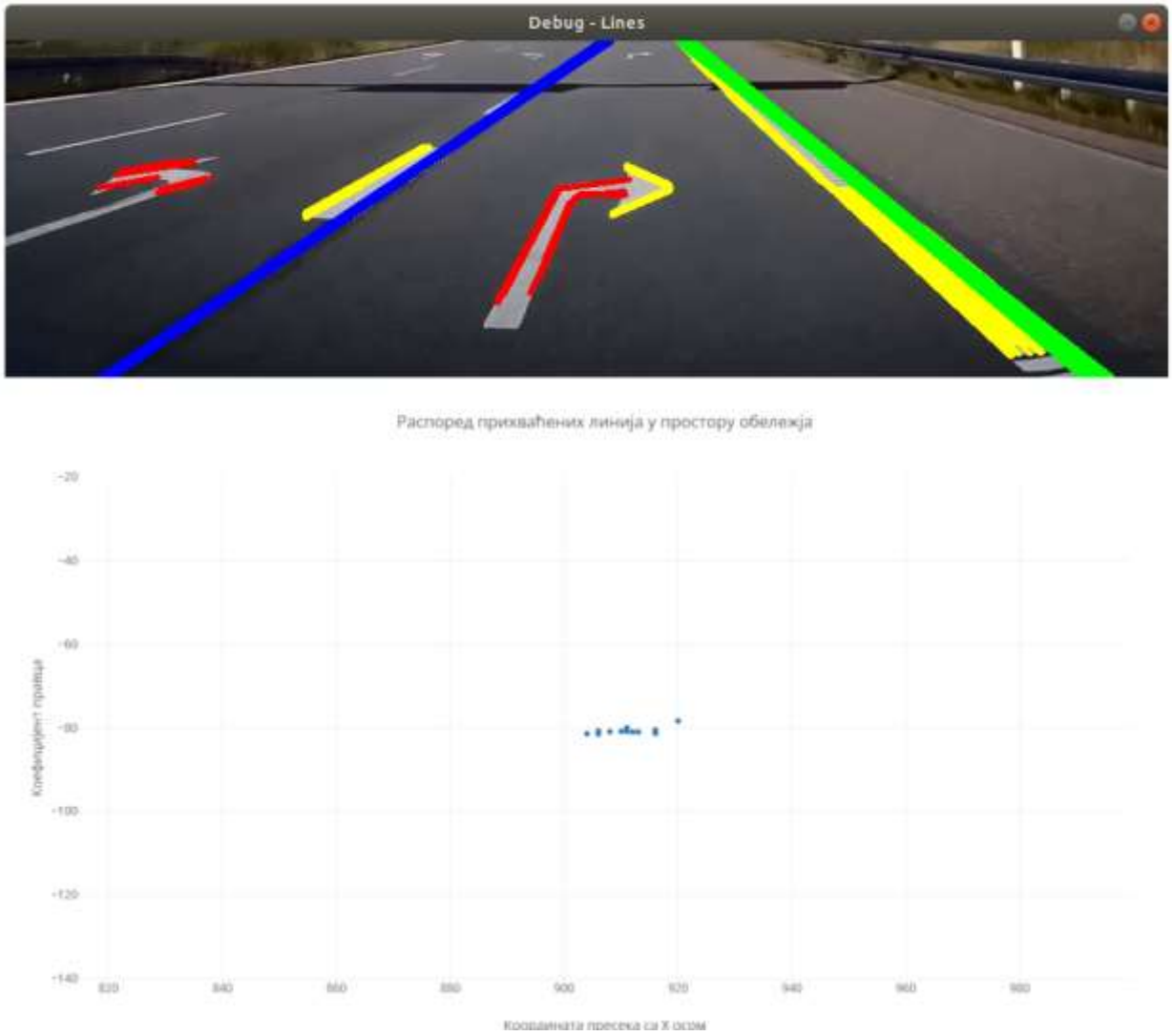
Последњи корак у класификацији је одбацивање преосталих линија шума и издвајање најбољег примерка обе од класа. Кључну улогу играју два обележја која су израчуната у претходним корацима, коефицијент правца и координата пресека са X осом. Ова два обележја формирају 2D простор где се свака права може представити као једна тачка. Да би распон вредсности био што приближнији, вредности коефицијента правца су увећане 200 пута. Очекујемо да ће се тачке нагомилавати у околини тачке чије вредности одговарају линији коју препознајемо. Очекујемо и да ће бити тачака које одступају од тог шаблона и те тачке представљају линије шума.

Слика 28 даје графички приказ расподеле у простору поменутих обележја. У горњем делу је слика која је резултат претходног разврставања линија, а испод ње налази се расподела тачака које представљају скуп десних линија са горње слике. На расподели се лако уочавају две групе тачака. Линије које су део стрелице на средини пута и које су по нагибу сврстане у скуп десних линија садасу се јасно издвојиле. Са слике можемо уочити да њихове пресечне тачке са X осом одговарају пресечним тачкама исправних линија, међутим, оне се значајно разликују по коефицијенту правца.



Слика 28: Расподела линија у простору обележја

Када смо уочили расподелу обележја, потребно је осмислити рачунарски механизам који ће одбацити тачке које имају велика одступања. Овај механизам одвија се у корацима. У сваком кораку одбацује се тачка која највише одступа од референтне тачке, која је одређена на основу просека свих тачака које се налазе у скупу. На положај референтне тачке, поред тренутног просека у скупу, утиче и положај тачке која је препозната као разделна линија на претходном фрејму. Референтна тачка се налази на половини линије између тренутног просека и издвојене тачке у претходном фрејму. Уколико у претходном фрејму линија није била препозната, тачка просека се узима за референтну тачку. Након одбацивања, механизам се понавља уколико постоји тачка која није унутар предефинисане области око референтне тачке. На овај начин, у сваком кораку смањујемо варијансу скупа. Слика 29 приказује расподелу прихваћених линија и слику на којој су одбачене линије приказане жутом бојом, а прихваћене плавом и зеленом.



Слика 29: Расподела прихваћених линија у простору обележја

Након одбацивања свих тачака које од референтне тачке одступају више од дозвољених вредности, можемо прерачунати коефицијент правца и тачку пресека са X осом. Ове две вредности рачунају се усредњавањем вредности које одговарају тачкама које су прихваћене.

Са одређивањем ове две вредности, готов је алгоритам препознавања. Коефицијент правца и пресечна тачка са X осом су довољан скуп података да се једнозначно одреди разделна линија. На крају, препозната линија се приказује на тренутном фрејму и прелази се на учитавање наредног фрејма и поновно примењивање целог алгоритма. Слика 30 приказује препознате разделне линије на фрејму који је служио као пример у објашњавању претходних корака.

Слика 30: Крајњи резултат обраде

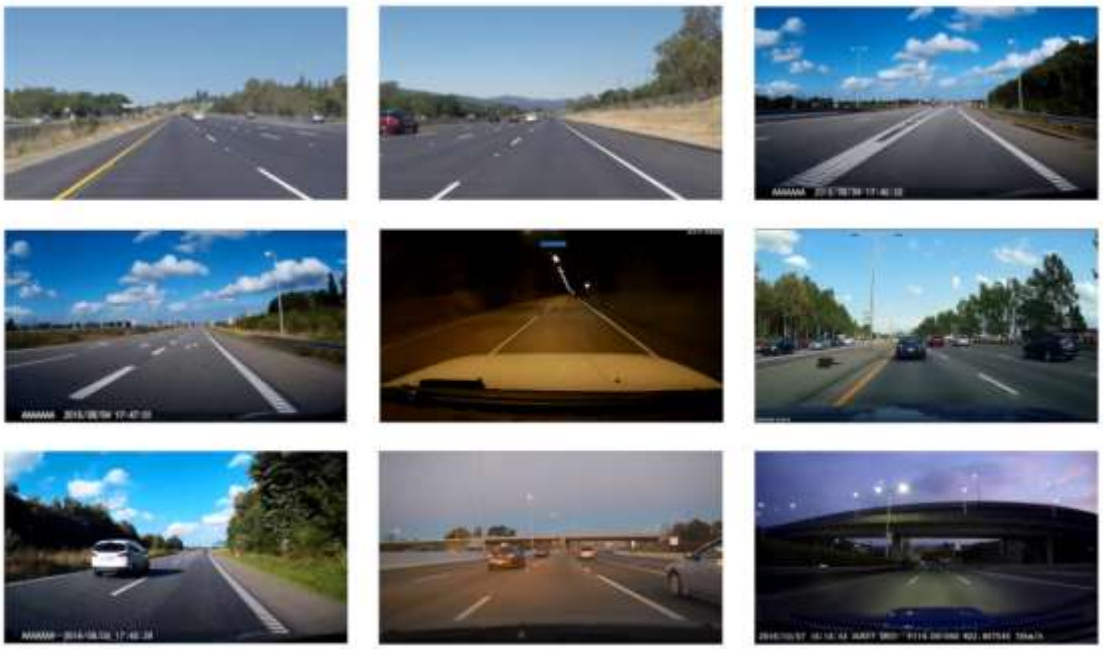
Алгоритам се кружно примењује све док има фрејмова тј. док се не дође до краја снимка. У реалној ситуацији, алгоритам би се примењивао до искључивања камере.

6. Испитивање решења

За време развоја, алгоритам смо испитивали аутоматским тестовима. Аутоматски тестови покретани су ручно након сваке веће промене. Њихова намена била је да осигурају стабилност развоја. Када би неки од испитних случајева падали, то би значило да је скорашња промена унела регресију. Аутоматско испитивање у спрези са механизмом за контролу верзија (енгл. *version control*) попут GIT-а омогућује лако праћење напретка пројекта и осигурава и стабилност развоја.

Развијени алгоритам се састоји од великог броја параметара. Вредности параметара одређиване су експерименталним путем, тј. променом вредности параметра, применом алгоритма и праћењем утицаја промене те вредности на успешност издвајања линије. Параметри су најчешће подешавани према једном од снимака. Често се дешавало да промена једног од параметара допринесе бољем издвајању линија на тренутном снимку, али наруши успешност препознавања линија алгоритма на другим снимцима. Аутоматски тестови су одиграли кључну улогу у брзом и ефикасном откривању поменутог сценарија.

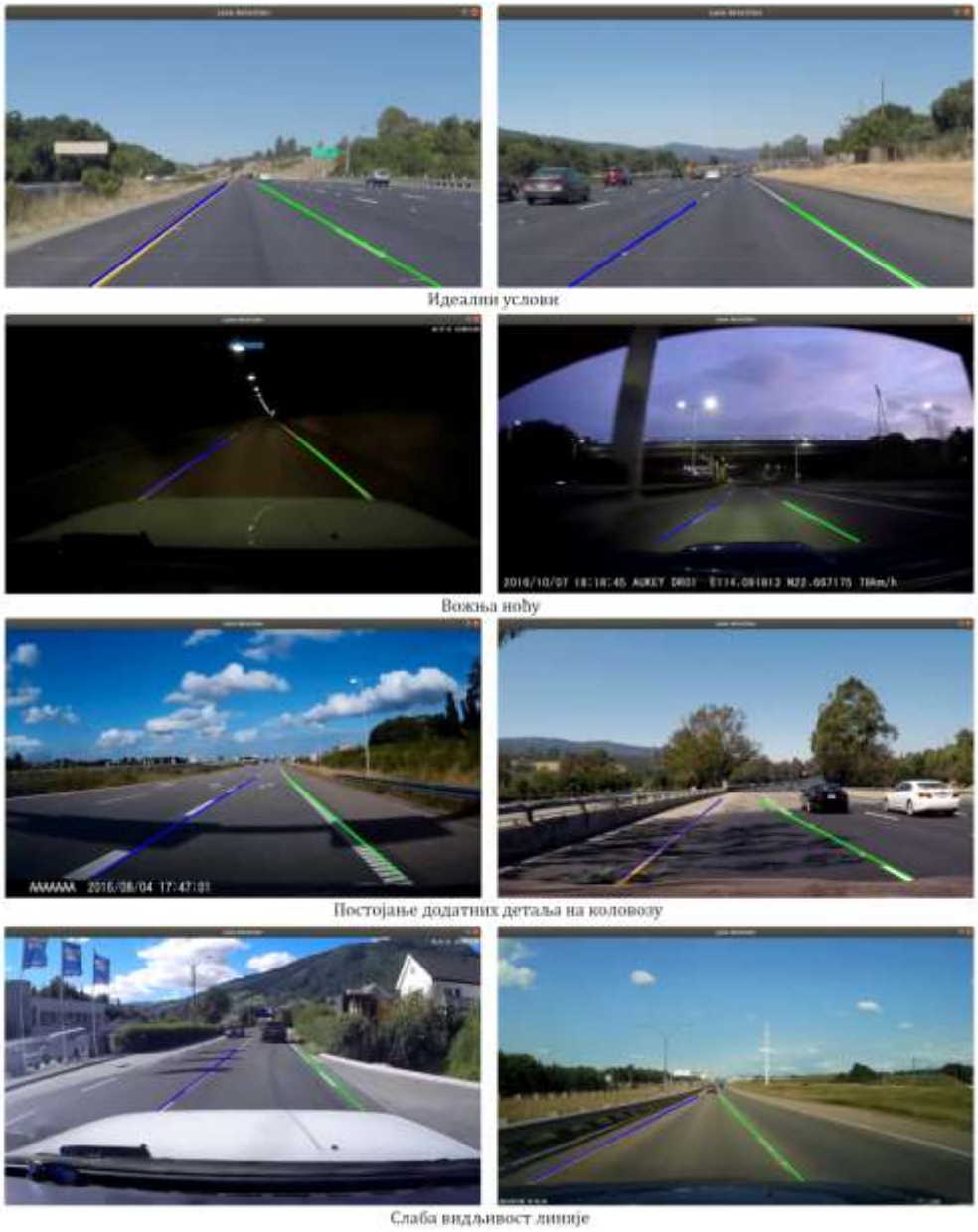
Аутоматски тестови написани су тако да примене тренутну верзију алгоритма на обраду девет различитих фрејмова. Испитни фрејмови су бирани тако да обухвате што више различитих сценарија. Тестови проверавају да ли је препознавање успешно и јесу ли препознате вредности у границама очекиваних. Тестови проверавају и колико времена је алгоритму потребно за обраду и да ли је то време у прихватљивим границама. На слици 31 приказани су испитни фрејмови.



Слика 31: Испитни фрејмови коришћени за аутоматско тестирање

7. Закључак

Развијени алогритам намењен је препознавању разделних линија коловозних трака на основу снимка камере постављене на ветробранско стакло. Алгоритам се успешно носи са постављеним задатком и може да издвоји разделне линије независно од доба дана и временских услова. Перформансе система су довољно добре да се обрада може одвијати у реалном времену, што значи да је време које је потребно за обраду једног фрејма мање од временског интервала између пристизања два фрејма. На слици 32 приказан је резултат препознавања на фрејмовима у различитим добима дана и различитим временским условима.



Слика 32: Резултати препознавања у различитим условима

На основу података које алгоритам обезбеђује, могуће је одредити релативни положај аутомобила у саобраћајној траци. Релативни положај аутомобила у саобраћајној траци може бити од користи систему који има задатак да аутономно управља аутомобилом и одржава га у саобраћајној траци, рецимо, током дуге вожње на ауто-путу. Податак о релативном положају аутомобила у односу на разделне линије саобраћајних трака може бити од користи и када је потребно препознати да ли аутомобил прелази из једне коловозне траке у другу.

Унапређење система могуће је направити у два правца, побољшањем поузданости и побољшањем перформанси. На побољшање поузданости може утицати увођење напреднијег метода класификације и додавање још једног обележја попут дужине издвојене линије. Увођењем додатног корака дигиталне обраде слике, који би се користио за поправљање контраста, знатно би се побољшала поузданост система у случајевима обраде недовољно осветљених слика. Побољшање перформанси може се постићи увођењем механизма који би сужавао област од интереса у зависности од области у којој су разделне линије препознате у претходном фрејму.

Недостатак система је повремена немогућност препознавања разделних линија на недовољно осветљеним сликама. Алгоритам тешко издваја разделне линије у случају када аутомобил пролази испод надвожњака или вози у сенци другог великог возила.

8. Литература

[1] *Global status report on road safety 2015* [2015]
Светска здравствена организација
apps.who.int/iris/bitstream/10665/189242/1/9789241565066_eng.pdf

[2] *AUTOMATED DRIVING SYSTEMS 2.0* [2017]
NHTSA, U.S. Department of Transportation, Сједињене америчке државе
nhtsa.gov/sites/nhtsa.dot.gov/files/documents/13069a-ds2.0_090617_v9a_tag.pdf

[3] *A flexible automotive systems architecture for next generation ADAS* [2018]
Johannes Hiltscher, Phanindra Akula, Robin Streiter, Gerd Wanielik
Chemnitz University of Technology, Беч, Аустрија

[4] *Questions and Answers about ABS* [2010]
https://bit.ly/2OUm797

[5] *THE ROLL STABILITY CONTROL ™ SYSTEM* [2011]
Jianbo Lu, Dave Messih, Albert Salib
Ford Motor Company, Сједињене америчке државе
https://bit.ly/2BvZvtH

[6] *Adaptive Cruise Control -Towards a Safer Driving Experience* [2012]
Rohan Kumar, Rajan Pathak
MIT College of Engineering, Пуне, Индија
https://bit.ly/2N2YhXX

[7] *A Real-World Multimodal Corpus to Monitor the Driver's Affective State* [2018]
http://www.adasandme.com/wp-content/uploads/2018/06/LREC_Lotz.pdf

[8] *About OpenCV* [2018]
OpenCV team
https://opencv.org/about.html

[9] *What Is the HSV Color Model* [2018]
Jacci Howard Bear
Lifewire, Сједињене америчке државе
https://www.lifewire.com/what-is-hsv-in-design-1078068

[10] *Mat - The Basic Image Container* [2018]
OpenCV team
https://bit.ly/2nViIeL

[11] *Digital Image Processing Third Edition* [2002], страница 741
Rafael C. Gonzales, Richard E.Woods
Pearson, Велика Британија

[12] *Canny Edge Detector* [2018]
OpenCV team
https://bit.ly/2AfyGJs

[13] *Digital Image Processing Third Edition* [2002], страница 755
Rafael C. Gonzales, Richard E.Woods
Pearson, Велика Британија

[14] *Hough Line Transform¶* [2018]
OpenCV team
https://bit.ly/2w0AEsN