Korištenje kamera kao zamjena za zrcalo u ADAS algoritmima

Zovak, Ivan

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

KORIŠTENJE KAMERA KAO ZAMJENA ZA ZRCALO U ADAS ALGORITMIMA

Diplomski rad

Ivan Zovak

Osijek, 2020.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 20.09.2020.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime studenta:	Ivan Zovak
Studij, smjer:	Diplomski sveučilišni studij Automobilsko računarstvo i komunikacije
Mat. br. studenta, godina upisa:	D-29ARK, 19.09.2019.
OIB studenta:	65005115541
Mentor:	Izv.prof.dr.sc. Ratko Grbić
Sumentor:	
Sumentor iz tvrtke:	Zvonimir Kaprocki
Predsjednik Povjerenstva:	Izv.prof.dr.sc. Tomislav Matić
Član Povjerenstva 1:	Izv.prof.dr.sc. Ratko Grbić
Član Povjerenstva 2:	Denis Vajak
Naslov diplomskog rada:	Korištenje kamera kao zamjena za zrcalo u ADAS algoritmima
Znanstvena grana rada:	**Obradba informacija (zn. polje računarstvo)**
Zadatak diplomskog rada:	Današnja vozila sadrže mnoštvo pomoćnih sustava koji vozaču i putnicima vožnju čine sigurnijom i ugodnijom. Cijena koštanja elektroničke opreme kao što su kamere, ekrani za prikaz itd. prilično se smanjila posljednjih desetak godina. Također, cijena računalne moći modernih procesora u automobilima svakodnevno pada. Sve to dovelo je između ostalog i do činjenice da gotovo svi automobili visoke i srednje klase, a većinom i oni niže klase, imaju implementirane kamere i velike ekrane za prikaz u automobilu, koji se često koriste i za prikaz stanja okoline automobila iz različitih perspektiva. U sklopu ovog diplomskog rada potrebno je na ekranu omogućiti prikaz slika s više različitih kamera
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	20.09.2020.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 24.09.2020.

Ime i prezime studenta:	Ivan Zovak
Studij:	Diplomski sveučilišni studij Automobilsko računarstvo i komunikacije
Mat. br. studenta, godina upisa:	D-29ARK, 19.09.2019.
Turnitin podudaranje [%]:	1

Ovom izjavom izjavljujem da je rad pod nazivom**: Korištenje kamera kao zamjena za zrcalo u ADAS algoritmima**

izrađen pod vodstvom mentora Izv.prof.dr.sc. Ratko Grbić

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD

Automobilska industrija jedna je od najbrže rastućih industrija današnjice. Razlog tome je automatizacija velikog broja funkcionalnosti koje je do sada obavljao vozač kako bi se vozaču olakšao proces vožnje, učinio je udobnijom i sigurnijom. Krajnji cilj je ostvarenje potpuno autonomne vožnje. Navedene funkcionalnosti ostvaruju se implementacijom različitih ADAS sustava (engl. *Advanced Driving Assistance Systems*). ADAS sustavi temelje se na računalnim sustavima koji dobivaju informacije o trenutnom stanju okoline vozila pomoću različitih senzora montiranih na vozilu ili unutar njega. Informacije koje se registriraju šalju se prema daljnjim programskim i sklopovskim elementima koji aktiviraju različite funkcije preko raznih aktuatora koji obavljaju zadane radnje, čime se postiže pomoć (ili potpuna zamjena) za vozača pri procesu vožnje.

ADAS sustavi dijele se na šest razina autonomije. Razine su određene brojevima od 0 do 5. Nulta razina podrazumijeva sustav koji nema niti jednu automatiziranu značajku, tj. vozač u potpunosti upravlja svim mehanizmima vozila. Na prvoj razini uvode se prvi mehanizmi koji pomažu vozaču u određenim načinima vožnje, kao što npr. sustav za održavanje konstantne brzine ili sustav za sprječavanje blokiranja kotača (engl. *Anti-lock Braking System* - ABS), ali je i dalje potrebna potpuna pozornost vozača. Druga razina podrazumijeva djelomičnu automatizaciju, pri čemu je sustav u barem jednom načinu rada autonoman – primjerice sustav koji održava brzinu ovisno o razmaku prema vozilu ispred (engl. *Adaptive Cruise Control*). Treća razina je trenutni stupanj razvoja koji je trenutno dozvoljen u prometu u Europi. Tu razinu odlikuje potpuna autonomnost vozila u odgovarajućim uvjetima prometa (npr. putovanje autocestom), ali vozač i dalje treba biti spreman preuzeti kontrolu nad vozilom u bilo kojem trenutku. Četvrta razina predstavlja gotovo potpunu autonomiju vozila, tj. vozač ne mora pratiti sustav, a sustav može kontrolirati cjelokupni proces vožnje, ali samo u određenim scenarijima (npr. određena geografska lokacija). Peta razina autonomije vozila predstavlja konačni stupanj autonomnosti vožnje, a podrazumijeva potpunu samostalnost računalnih sustava u vozilu u kojima primjerice neće biti potrebni volan i papučice za kočenje i davanje brzine. Kod pete razine intervencija vozača uopće nije potrebna. Posljednje dvije razine još nisu puštene u promet, ali se radi na njihovim implementacijama i testiranjima [1] .

Uobičajeni senzor za percepciju okoline u ADAS sustavima je kamera. Upravo su kamere korištene u ovom radu kao senzori za detekciju vozila iza vozila na kojem kamera stoji. Daljnje primjene mogu biti: detekcija pješaka, znakova, linija na prometnicama, pomoć pri parkiranju

vozila, panoramski pogled okoline vozila i slično. Osim kamera koriste se i senzori koji pomoću radiovalova (engl. *Radio Detection And Ranging* - RADAR) i svjetlosnih valova (engl. *Light Detection And Ranging* - LIDAR) detektiraju prepreke, ali i mapiraju okolinu vozila. Princip rada zasniva se na mjerenju vremena potrebnog da se odašiljana zraka vrati nakon odbijanja. Jedan od načina dobivanja informacija o stanju okoline vozila je i razmjena podataka između vozila putem odgovarajuće mrežne infrastrukture čime je moguće dodatno unaprijediti ADAS sustave. Međutim, kako bi se složeniji ADAS sustavi mogli koristiti u prometu, potrebno je unaprijediti cijelu trenutnu infrastrukturu prometa – urediti prometnice i postaviti preusmjerivače signala dovoljno gusto da se može odvijati komunikacija između vozila i infrastrukture.

ADAS sustavi se trenutno ne ugrađuju samo u vozila visoke klase, već i u vozila srednje klase. Neki od sustava koji se ugrađuju su: kamera za vožnju unatrag ili pomoć pri parkiranju, sustavi koji kamerama zamjenjuju zrcala, sustav koji analizira mogućnost pretjecanja, sustav za upozoravanje o napuštanju vozne trake, sustav za prepoznavanje znakova, sustav za održavanje konstantne brzine, sustav koji ograničava brzinu vožnje, sustav koji održava razmak od vozila koje se nalazi ispred vlastitog vozila itd. Implementaciju svih tih sustava ograničavaju brojni zahtjevi, kao npr. ograničenje prostora u vozilima (posebice u automobilima). Isto tako, velika ograničenost manifestira se u zahtjevima za računalnom moći – količini memorije i brzini procesora. Najveći izazov autonomne vožnje je izvedba sustava u realnom vremenu (engl. *real-time*). Odziv za izvođenje u realnom vremenu treba biti otprilike manji od 300 milisekundi (ovisno o primjeni), a taj zahtjev je nužan posebno za sigurnosno-kritične sustave poput sigurnosnog pojasa, ABS-a, zračnih jastuka, detekcije prepreka i sl [2]. Primjerice, kašnjenje od samo pola sekunde za otvaranje zračnih jastuka može rezultirati kobnim neželjenim posljedicama. S druge strane, izvođenje u realnom vremenu nije toliko bitno kod informacijskih sustava poput informativno-zabavnog ekrana (engl. *infotainment*).

U ovom radu su predloženi i implementirani algoritmi koji pomoću kamera obavljaju funkcionalnost zrcala u vozilima. Uz sami prikaz slike dodane su funkcionalnosti za zrcaljenje i uvećavanje slike, te za detekciju vozila i izračunavanje udaljenosti i brzine. Prikazan je cijeli redoslijed algoritma – od dobivanja signala s kamera preko obrade slike do prikazivanja na ekran. Predloženi algoritam implementiran je za ugradbenu računalnu platformu ADAS Alpha koja ima mogućnost spajanja većeg broja automotiv kamera. Ulaz u algoritam je signal s više kamera, pri čemu su tri kamere širokokutne (engl. *fisheye*), a dvije kamere uskokutne (engl. *narrow angle*). Sve navedene funkcionalnosti raspoređene su u ukupno 5 korisničkih slučajeva s mogućnostima korisničkog odabira nekih varijabilnih dijelova, npr. dijela slike za uvećavanje. Cjelokupno

rješenje razvijeno je u programskom jeziku C unutar programskog okruženja VISION SDK, pri čemu je posebni naglasak na što efikasnijem izvođenju programskog kôda – raspoređivanje računalnih operacija na više procesora, optimizirano kodiranje i slično. Predloženo rješenje verificirano je ispitivanjem postignutih funkcionalnosti i točnosti te mjerenjem FPS-a i opterećenjem procesora.

Rad je strukturiran na sljedeći način. Drugo poglavlje rada daje pregled postojećih rješenja u automobilskoj industriji. Prikazani su različiti proizvođači automobila i njihove specifikacije za zamjenu zrcala pomoću kamera. U trećem poglavlju predstavljeno je predloženo rješenje korištenja kamera kao zamjena za zrcalo u ADAS algoritmima, opisano je dostupno razvojno okruženje VISION SDK i ADAS Alpha ploča. U četvrtom poglavlju navedeni su rezultati verifikacije rješenja, a peto poglavlje sadrži zaključak.

2. KAMERE KAO ZAMJENA ZA ZRCALO

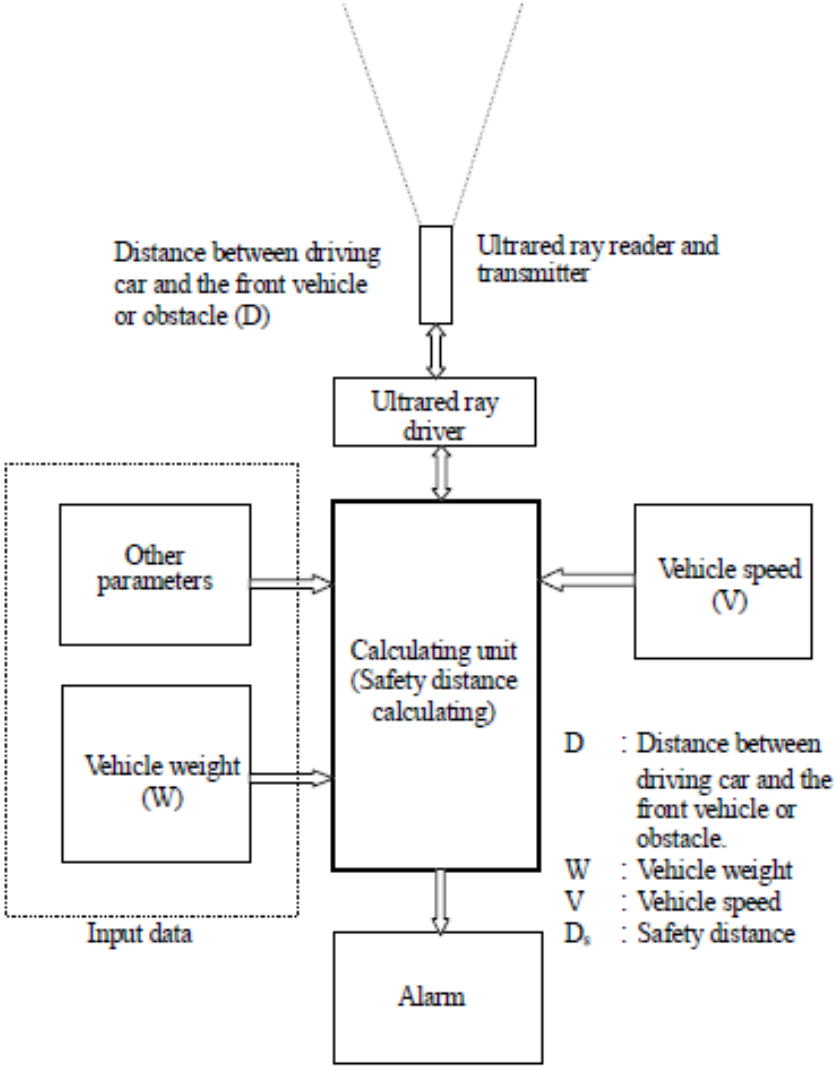
U posljednjih nekoliko godina kamere se sve više ugrađuju u vozila. Ugrađuju se na prednjoj, stražnjoj i bočnim stranama vozila. Koriste se i širokokutne i uskokutne kamere. Primjerice, u SAD-u , sva vozila prodana nakon svibnja 2018. čija je cijena ispod 10 000 dolara morala su imati zadnju kameru. S druge strane, u Europi Euro-NCAP ocjene zahtijevaju sve više sigurnosnih značajki od proizvođača od 2016. godine kako bi održali željenu ocjenu klase vozila. To rezultira dodavanjem sigurnosnih sustava čak i u vozila srednje ili niže klase, što dovodi do više inovacija i smanjenje cijene proizvodnje ovakvih tehnologija [3] .

2.1. Pregled nekih od ADAS sustava temeljenih na kameri kao senzoru

U ovom potpoglavlju opisane su različite primjene kamera kao senzora za ADAS algoritme - prikaz stanja okoline oko vozila, izračunavanja udaljenosti i upozoravanje vozača. Tvrtka Texas Instruments kreira proizvode pogodne za pokretanje algoritama koji se koriste u ADAS sustavima. Najpoznatija grupa procesora je TDA2xx sustav na čipu (engl. *System on Chip* – SoC) koja sadrži procesore posebno dizajnirane za obradu signala kamera montiranih na prednjoj i stražnjoj strani vozila. U radu [4] prikazano je korištenje širokokutne kamere ovog proizvođača. Ona omogućuje pregled puno šireg kuta vidljivosti (engl. *Field of View* - FoV) za razliku od uskokutne kamere. Međutim, širokokutna kamera sadrži značajno izobličenje, tj. distorziju realne scene koja se treba ispraviti kako bi se slika mogla nadalje obrađivati. Isti tip kamera koristi se i za pogled odozgor (ptičja perspektiva) na vozilo (engl. S*urround View*). Time se postiže vidljivost prostora oko vozila (360 stupnjeva), a sustav sadrži 4 do 6 kamera kojima se kutovi vidljivosti preklapaju. Kako bi dobio se smisleni prikaz okoline, potrebno je provesti 2 procesa: geometrijsko i fotometrijsko poravnanje [5]. Geometrijsko poravnanje podrazumijeva povezivanje susjednih slika tako da se detektiraju isti elementi u obje slike i prikažu kao jedan element, pri čemu ne smije biti vidljivog diskontinuiteta. Nakon toga, zbog različitog osvjetljenja svake kamere, osvjetljenje svakog objekta može biti drugačije na svakoj kameri. Fotometrijsko poravnanje (detaljnije opisano u [6]) služi za ujednačavanje osvjetljenja i boja, tako da slike s različitih kamera izgledaju kao slika s jedne kamere (engl. *Stitching*). To se radi pomoću globalne funkcije za korekciju boja i svjetline u preklapajućim područjima oko vozila.

U radu [7] prikazuje se dobivanje slika visoke kvalitete iz izvornog (engl. *raw*) formata s kamera u ADAS sustavima na TDA3x seriji procesora pomoću raznih tehnika obrade slike. Opisan je algoritam za detekciju prepreka i izračun za sigurnosnu udaljenost. Prikazan je izračun udaljenosti vozila od prepreke ispred vozila koristeći čitač i transmiter infracrvenih zraka. Uzimaju

se u obzir brzina vozila, masa vozila, vremenski uvjeti i nagib ceste. Ovisno o tim parametrima, modul za izračunavanje određuje sigurnosnu udaljenost pri kojoj se vozilo može sigurno zaustaviti. Poznavajući sigurnosnu udaljenost, sustav tu varijablu uspoređuje s udaljenosti od prepreke. Ako je sigurnosna udaljenost veća od udaljenosti do prepreke, vozaču se aktivira alarm. Na slici 2.1. prikazana je shema sustava. U radu je zaključeno kako je funkcija puta kočenja i brzine kojom se vozilo kreće približno linearna. Isto vrijedi i za ovisnost vremena kočenja o brzini kojom se vozilo kreće. Obje funkcije zavise o učinkovitosti kočnica, gustoći zraka, otporu zraka faktoru trenja i faktoru kotrljanja.



Sl. 2.1. Sustav za upozoravanje o sigurnoj udaljenosti. [7]

Rad [8] također prikazuje metodu za detekciju vozila i procjenu udaljenosti u okolini autonomnog vozila, pri čemu se vozaču šalje upozorenje o potencijalnom sudaru. Implementacija ovog rada izvršena je u programskom okruženju ROS pomoću kamera (engl. *Robot Operating System*), prvenstveno kao alternativa sustavima s RADAR i LiDAR tehnologijama. Pomoću YOLO v2 algoritma implementirano je prepoznavanje 4 vrste vozila: automobil, kombi, kamion i autobus. U ovom radu su za procjenu udaljenosti korištena dva ROS čvora. Jedan služi za procjenu udaljenosti u Carla simulacijskom programu, a drugi čvor za procjenu udaljenosti u stvarnom svijetu. Za učenje mreže korištene su slike iz više izvora: javno dostupne baze podataka i samostalno snimljeni video materijali s mobilnog telefona.

2.2. Primjeri kamera kao zamjena za zrcalo u automobilskoj industriji

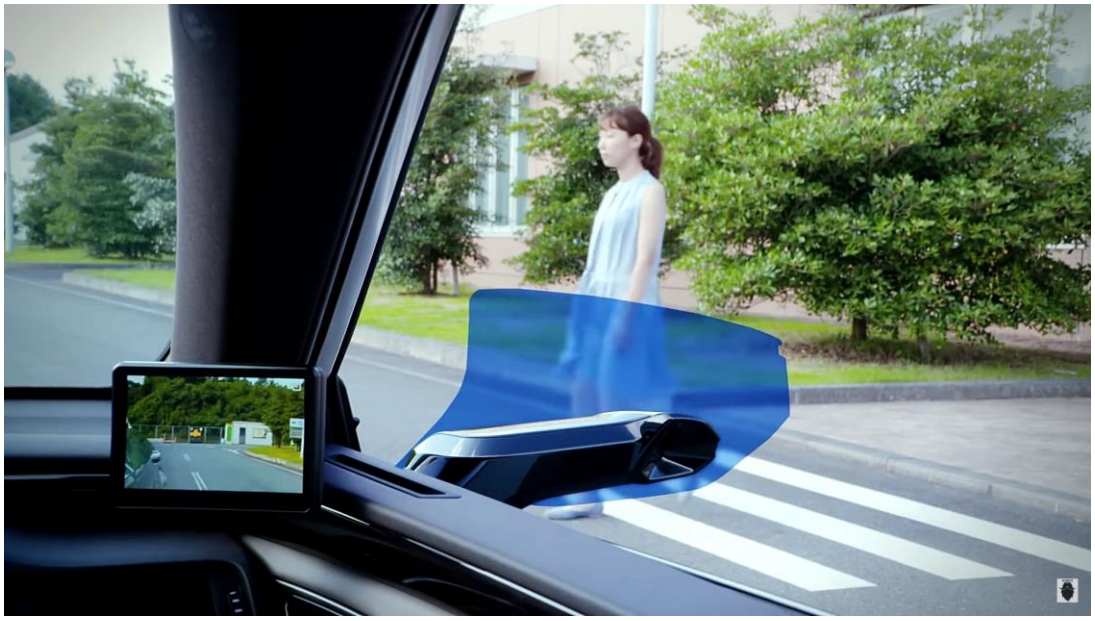
U ovom potpoglavlju navedeni su primjeri korištenja kamera kao zamjena za zrcalo u automobilskoj industriji. Prvi primjer je BMW i8 eDrive iz 2016. godine. Na slici 2.2. [9] vidi se kompletni prikaz stražnjeg dijela automobila na poziciji konvencionalnog središnjeg zrcala u kabini vozila, dobiven preciznim homogenim spajanjem slika sa 3 kamere. Na sredinu tog prikaza dodana je virtualna grafička ilustracija samog automobila. Uz takav pregled vozač se pri vožnji unatrag uopće ne mora okretati, dovoljno je samo gledati u središnji ekran. Ako se neko vozilo nalazi u blizini, pojavljuje se znak upozorenja. Također, na promotivnom videu vidi se kako se pri skretanju dio scene koji je bliže zavoju dodatno proširuje, što je pogodno primjerice za pješake ili bicikliste koji se mogu pojaviti u području mrtvog kuta. Dodatna funkcionalnost je dojava vlasniku automobila u slučaju da je došlo do oštećenja karoserije, tako da se na vozačevom mobitelu uživo prikaže okolina vozila s kamera.

Sl. 2.2. Prikaz zrcala automobila - BMW i8 eDrive.

Audi e-tron iz 2019. godine ima ugrađen sustav koji prema brzini kojom se vozilo kreće mijenja na ekranu prikaz okoline iza automobila. Primjerice, ako se vozilo kreće autocestom brzinom 130 kn/h, vozaču je prikazana šira scena nego kada se vozilo kreće po gradu brzinom 50 km/h. Na ekranu prikazanom na slici 2.3. [10] moguće je prstom kliznuti i odabrati dio scene koji se želi prikazati (npr. pri parkiranju je poželjno vidjeti dio uz rub parkirnog mjesta kako se ne bi oštetile gume automobila). Dobru vidljivost osigurava i to što je kamera natkrivena, pa kiša ili druge vremenske neprilike ne mogu učiniti prikaz lošijim kao kod konvencionalnog zrcala u automobilu. Dodatna funkcionalnost je informiranje vozača o sceni iza vozila – ako se približava drugo vozilo prikazuje se upozorenje na zaslonu. Ako vozač uključi signalizaciju za prestrojavanje, a to nije moguće zbog nadolazećeg vozila, sustav prikazuje dodatno upozorenje i ne dozvoljava okretanje upravljača, tj. skretanje automobila.

Sl. 2.3. Prikaz ekrana automobila Audi e-tron.

Još jedan primjer iz automobilske industrije je Lexus ES 300h iz 2020. godine. Na slici 2.4. [11] vidi se razlika izgleda scene iz automobila s konvencionalnim zrcalom i s kamerom. S fizikalne strane, zamjenom zrcala smanjen je i otpor zraka i povećana je vidljivost. Lexus također ima i noćni način rada koji se automatski aktivira kad se promijeni osvjetljenje okoline automobila. Na slici 2.5. [11] prikazana je vidljivost okoline automobila pri lošim vremenskim uvjetima.



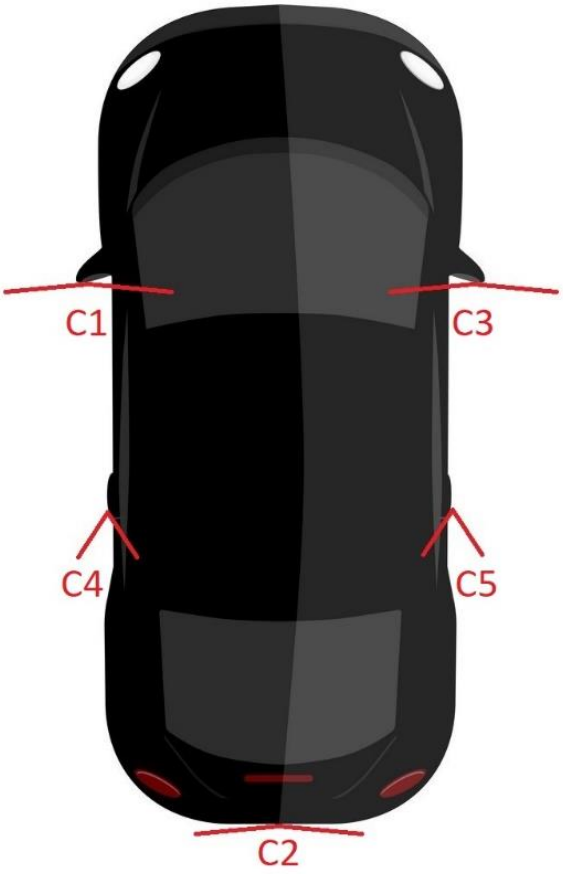
Sl. 2.4. Prikaz razlike konvencionalnog i modernog zrcala – Lexus ES 300h.

Sl. 2.5. Prikaz ekrana pri lošim vremenskim uvjetima – Lexus ES 300h.

Najveća prednost korištenja kamera kao zamjenu za zrcala je zasigurno veći FoV te bolja vidljivost u slučaju loših vremenskih uvjeta (npr. kiše). Uz to, velika prednost su i sustavi obavještavanja vozača koji ne postoje kod konvencionalnih zrcala. Mogući nedostatak kamera može biti neoptimiziranost i nepouzdanost sustava, što može dovesti do neželjenih posljedica za vozača i vozilo.

3. PREDLOŽENO RJEŠENJE ZA KORIŠTENJE KAMERE KAO ZAMJENE ZA ZRCALO U VOZILIMA

U ovom poglavlju predloženo je rješenje koje omogućava korištenje kamera kao zamjene za zrcalo. Rješenje se temelje na maketi vozila, ADAS ploči i kamerama. Na slici 3.1. prikazan je raspored pozicija montaže kamera na maketu vozila, a ilustrirani su i tipovi kamera. Korištene su 3 širokokutne i 2 uskokutne kamere. Kamere su raspoređene tako da pokriju cijelo područje koje se nalazi bočno ili iza vozača. Time se osigurava potpuni gotovo potpuni pregled vozila – sve osim područja ispred vozača. Predloženo rješenje implementirano je na jedan od SoC-eva koji se nalazi na ploči ADAS Alpha AMV. Za razvoj rješenja korišteno je programsko okruženje VISION SDK. Rješenje mora osigurati sljedeće: prikaz okoline oko vozila sa maksimalnim kutom vidljivosti, prikaz okoline za pomoć pri parkiranju, prikaz okoline sa mogućnosti uvećanja željenog dijela slike, detekciju nadolazećeg vozila, procjenu udaljenosti i brzine detektiranog vozila.

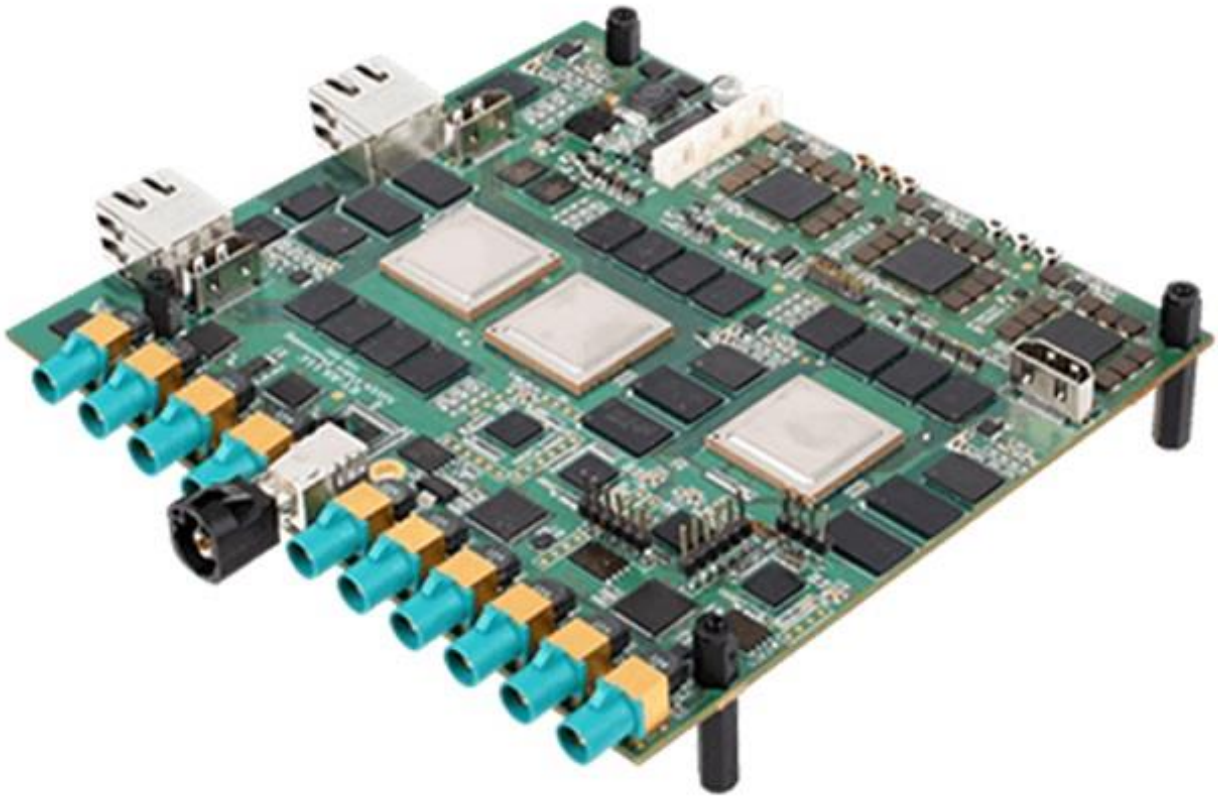


Sl. 3.1. Prikaz svih kamera na vozilu.

3.1. Opis ADAS Alpha platforme i VISION SDK okruženja

ADAS Alpha razvojna ploča, prikazana na slici 3.2., namijenjena je za implementaciju ADAS algoritama i korištena je u ovom radu. Sadrži tri SoC-a, a svaki se koristi za različite operacije. Nazivi SoC-eva su SC, FFN i FUS. Svaki SoC sadrži nekoliko različitih procesorskih jezgri (korisnik može raspoređivati zadatke između dva DSP, dva A15, dva M4 i 4 EVE procesora – pri čemu se razlikuju po radnim frekvencijama i namjeni), akceleratore, utore za microSD kartice, RAM, *flash* memoriju, izlaze i ulaze. Također, svaki se može pokrenuti u tri različita načina rada: Debug, SD i QSPI mod. Preko microSD kartice se na ploču prenose program pomoću kojeg se pokreće određeni SoC. Neke od mogućih funkcionalnosti su:

- Pogled oko vozila
- Prepoznavanje prometnih znakova
- Pomoć pri parkiranju
- Pogled unatrag
- Praćenje vozačevog lica i slično

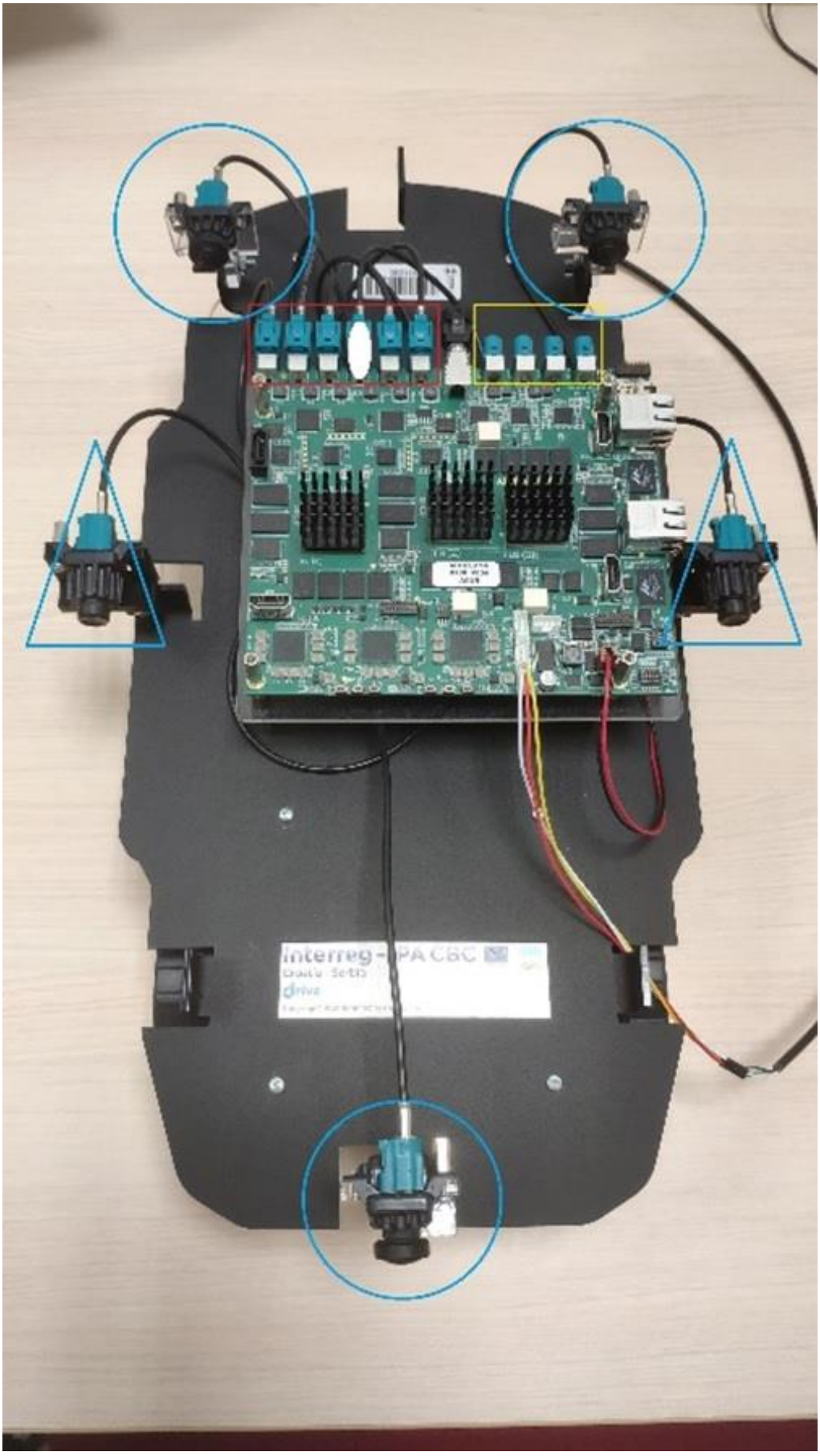


Sl. 3.2. ADAS Alpha ploča.

VISION SDK je programsko okruženje pogodno za ADAS ploče kreirane od strane tvrtke Texas Instruments. Unutar VISION SDK okruženja moguće je kreirati razne tokove podataka za ADAS korisničke slučajeve, obrađivati ih na različitim procesorskim jedinicama koristeći postojeće sklopovske akceleratore, a sve s ciljem optimizirane upotrebe ugrađenih SoC-eva. Bitna značajka je da VISION SDK okruženje dolazi s unaprijed kreiranim korisničkim slučajevima koji korisniku demonstriraju mogućnosti korištenja ADAS Alpha ploče i samog programskog okruženja. VISION SDK također omogućuje snimanje i preuzimanje videa, algoritme za kompresiju, prikaz i analizu videa.

Korisnik može samostalno raspoređivati i pozivati dijelove API-ja željenim i smislenim redoslijedom – u svrhu izvođenja nekog algoritma. Također, VISION SDK ne nudi automatsko raspoređivanje opterećenja pojedinih procesora, kao i odabir vrste procesora pogodnog za zadani algoritam. Osnovna ideja VISION SDK okvira je okvir *Links and Chains,* dok se programsko sučelje naziva *Link API.* Instalacijski paket sadrži i sve alate i elemente nužne za prevođenje aplikacija.

Na slici 3.3. prikazana je maketa vozila koja je korištena prilikom izrade ovog rada. Crvenom bojom uokvireni su konektori kamera SC SoC-a na koji su spojene kamere koje se koriste u radu. Bijelom bojom naznačena je kamera koja je spojena a ne koristi se. Žutom bojom uokvireni su konektori kamera FFN SoC-a koji nije korišten za potrebe ovog rada. Krugovima su označene širokokutne kamere, a trokutima uskokutne.

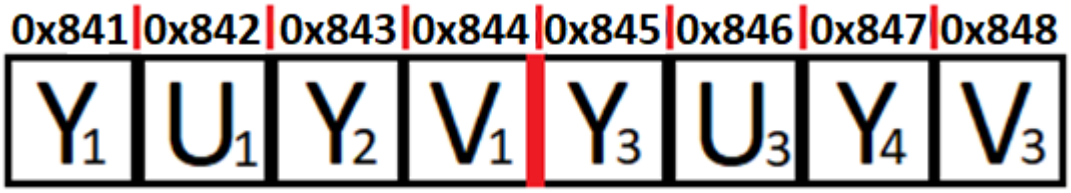


Sl. 3.3. Maketa vozila s montiranom ADAS pločom i kamerama.

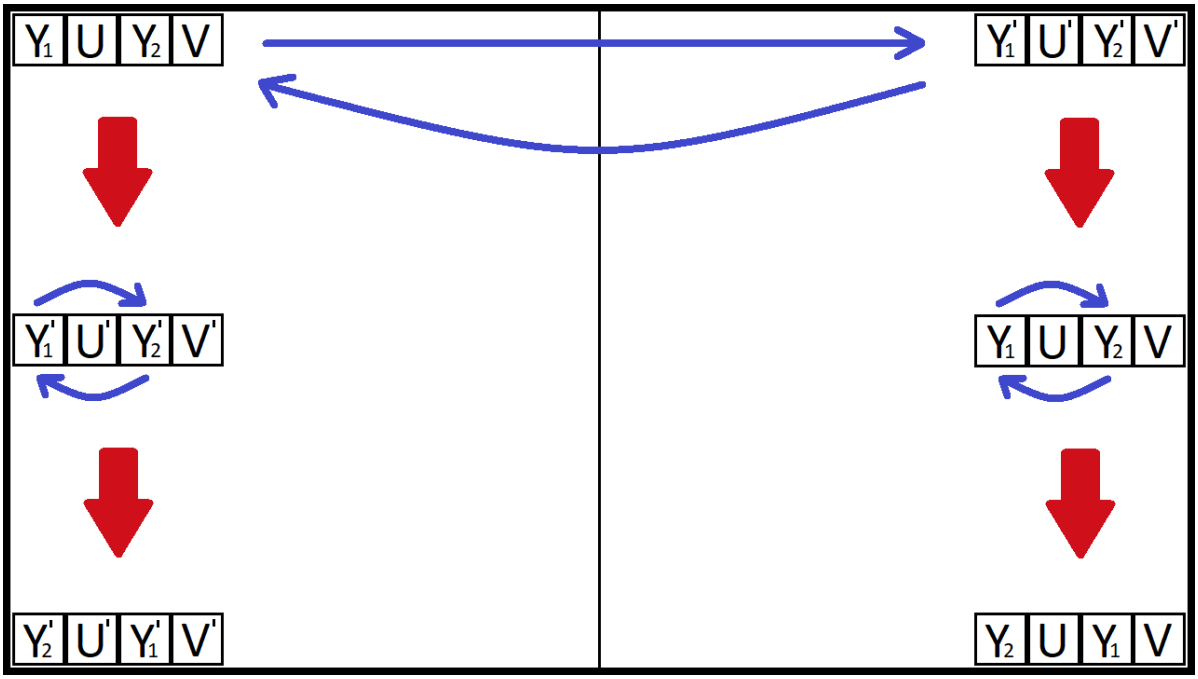
3.2. Algoritam za zrcaljenje slike

Kako bi se što vjerodostojnije korisniku prikazala scena iza vozila, potrebno je modificirati signal s kamera. Cilj je da objekti koji su s npr. desne strane vozila i na slici budu na desnoj strani (jednako kao i korištenjem zrcala). U ovom potpoglavlju predstavljen je algoritam za zrcaljenje slike (*Mirror*) koji je implementiran u obliku algoritamskog *linka*. Odlika algoritamskog *linka* je mogućnost korištenja u bilo kojem korisničkom slučaju. Ostvarenje tog zahtjeva osmišljeno je na način da slika sa svake kamere bude zrcaljena. Točnije, u svakom trenutku se pikseli zamjenjuju u odnosu na vertikalnu središnju os slike. U svim korisničkim slučajevima korišten je YUV format boja sa shemom poduzorkovanja 4:2:2. Y predstavlja svjetlosnu komponentu, a U i V su komponente boje određenog elementa slike. 4:2:2 shema poduzorkovanja smanjuje količinu podataka za trećinu (za svake 4 Y komponente umjesto 8 U i V komponenti koriste se 4 komponente boje). Dakle, svaka dva elementa slike u nizu imat će istu komponentu boje (isti U i V). Na slici 3.4. prikazan je način zapisa YUV formata sa poduzorkovanjem 4:2:2 u memoriji. Svaki element slike prikazan je s 32 bita (svaka komponenta po 8 bitova). Upravo zbog načina zapisivanja u memoriji nije dovoljno samo zamijeniti elemente slike, tj. prebaciti ih simetrično s obzirom na vertikalnu središnju os slike, nego je potrebno i unutar svakog elementa slike zamijeniti Y komponente. Princip rada algoritma zrcaljenja slike na primjeru dva elementa slike prikazana je na slici 3.5.

Programski kôd koji izvršava spomenutu zamjenu implementiran je u algoritamski *link Mirror* i prikazan je na slici 3.6. Kôd je osmišljen kao dvostruka *for* petlja koja prolazi kroz sve redove slike i pola stupaca slike (iz razloga što se algoritam izvršava s obzirom na vertikalnu središnju os slike). U linijama 5,6 i 7 vrši se zamjena dvaju elemenata slike korištenjem pomoćne varijable. U linijama 9,10,17 i 18 iz elementa slike izlučuju se Y komponente, uzimajući u obzir zapis formata u memoriji. U linijama 11,12,19 i 20 se te Y komponente pomjeraju na mjesto druge Y komponente (pomak za 16 mjesta u binarnom, tj. za 4 mjesta u heksadekadskom zapisu), tj. zamjenjuju se pozicije Y komponenti u jednom elementu slike. Zatim se u linijama 13 i 21 na originalne elemente slike dodaju modificirane maske gdje su Y komponente 0, a U i V komponente onakve kakve jesu. Na kraju se u linijama 14,15,22 i 23 na originalne elemente slike dodaju varijable u kojima su Y komponente zamijenjene.



Sl. 3.4. Zapis YUV 4:2:2 formata u memoriji.



Sl. 3.5. Princip rada algoritma za zrcaljenje slike.

Linija Kod

```
1:      for (rowIdx = 0; rowIdx < height; rowIdx++) {
2:
3:        for (colIdx = 0; colIdx < wordWidth / 2; colIdx++) {
4:
5:          temp = * (inputPtr + colIdx);
6:          *(inputPtr + colIdx) = * (outputPtr - colIdx);
7:          *(outputPtr - colIdx) = temp;
8:
9:          temp = * (inputPtr + colIdx) & 0x00FF0000;
10:         temp1 = * (inputPtr + colIdx) & 0x000000FF;
11:         temp = temp >> 16;
12:         temp1 = temp1 << 16;
13:         *(inputPtr + colIdx) = * (inputPtr + colIdx) & 0xFF00FF00;
14:         *(inputPtr + colIdx) = * (inputPtr + colIdx) | temp;
15:         *(inputPtr + colIdx) = * (inputPtr + colIdx) | temp1;
16:
17:         temp = * (outputPtr - colIdx) & 0x00FF0000;
18:         temp1 = * (outputPtr - colIdx) & 0x000000FF;
19:         temp = temp >> 16;
20:         temp1 = temp1 << 16;
21:         *(outputPtr - colIdx) = * (outputPtr - colIdx) & 0xFF00FF00;
22:         *(outputPtr - colIdx) = * (outputPtr - colIdx) | temp;
23:         *(outputPtr - colIdx) = * (outputPtr - colIdx) | temp1;
24:
25:       }
26:       inputPtr += (inPitch[0] >> 2);
27:       outputPtr += (inPitch[0] >> 2);
28:     }
```

Sl. 3.6. Kôd algoritma za zrcaljenje slike.

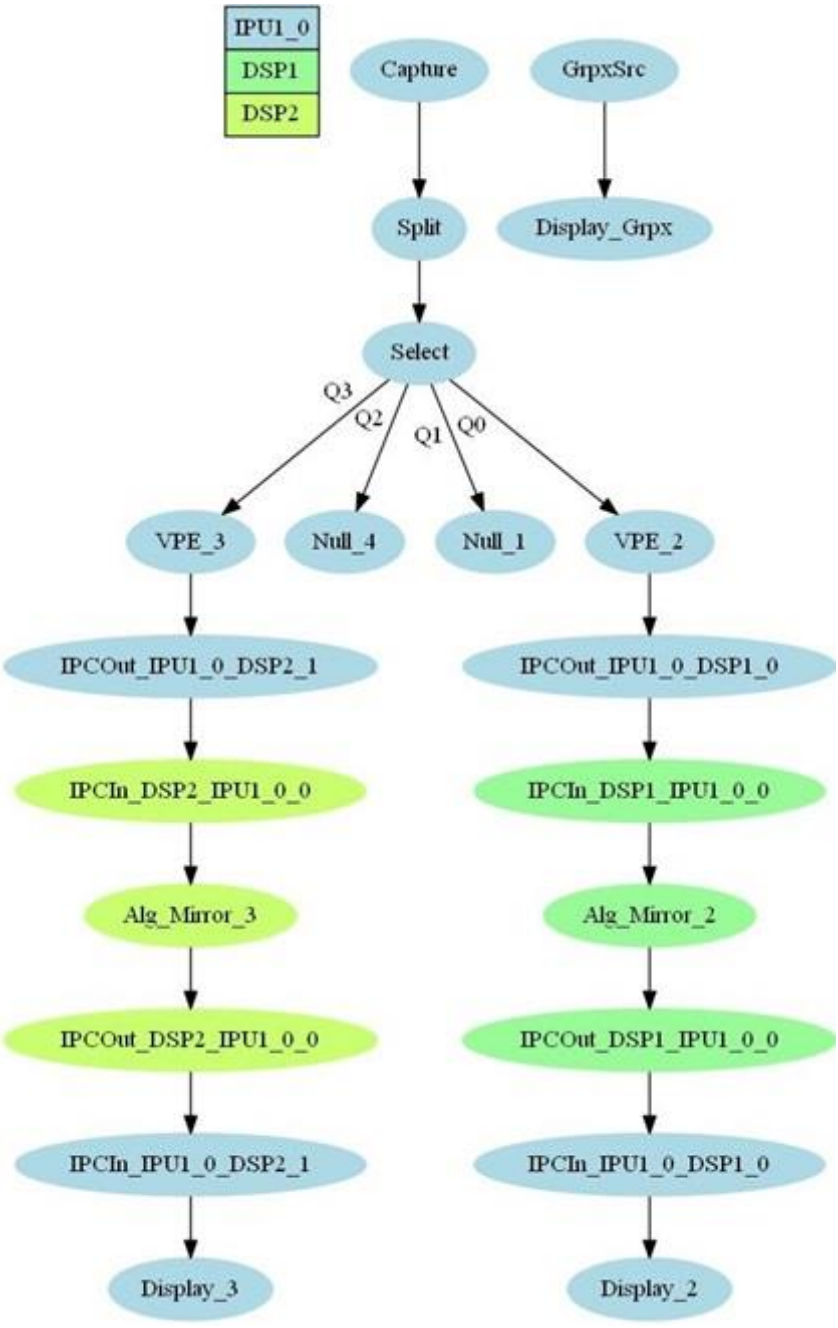
3.3. Korisnički slučaj s dvije kamere

Korisnički slučaj s dvije kamere osmišljen je kako bi se vozaču dodatno omogućio pregled okoline vozila. Koriste se jedna širokokutna i jedna uskokutna kamera. To može biti korisno pri parkiranju vozila, jer se omogućuje dodatni pregled okoline vozila. Rezolucija signala s kamera korištenih u ovom radu je 1280 x 720 elemenata slike, a u izlaznom signalu povećana je na 1920 x 1080 elemenata slike. Na slici 3.7. je prikaz korisničkog slučaja s dvije kamere u kojem se konstantno izvršava algoritam *Mirror* i ostali potrebni *linkovi* koji predstavljaju određene programske blokove. Svaki *link* određen je ulaznim i izlaznim brojem redova koji sadrže signale (kanale).

Capture link vrši dohvaćanje signala s kamera. Pri korištenju SC SoC-a, u konfiguraciji *Capture linka* potrebno je odrediti broj redova i signala (svaki signal sadrži dvije kamere koje su u paru – može se koristiti maksimalno šest kamera). *Split link* razdvaja ulazne signale u jednom redu na dva ili više podkanala u jednom redu (ovisno o broju kamera, signal veće rezolucije dijeli na više signala manje rezolucije). *Select link* dijeli jedan red na više izlaznih redova. U programskom kodu *Select linka* može se odrediti raspored i odabir kanala. U ovom korisničkom slučaju to je korisno jer se koriste dvije kamere, a s *Capture linka* poslani su signali s 4 kamere. Stoga se dva signala koji su nepotrebni šalju na *Null linkove* (koji zapravo nemaju funkciju), dok se preostala 2 upotrebljiva kanala raspoređuju po želji korisnika. U *VPE linku* može se odabrati format slike (u ovom slučaju izabran je YUV 4:2:2), slika se može skalirati po širini i visini, pozicionirati na ekran i izrezati po želji. Upravo je u VPE *linku* rezolucija ulaznog signala povećana sa 1280 x 720 elemenata slike na 1920 x 1080 elemenata slike. Na svakoj kameri konstantno se izvršava algoritam *Mirror* (opisan u potpoglavlju 3.2), pri čemu je potrebno obratiti pozornost na podjelu rada na više procesorskih jedinica. Nakon isprobavanja pokretanja algoritma na svim procesorima, zaključeno je kako operacije koje provodi algoritam najpogodnije odrađuje DSP grupa procesora. Posljednji *link* je *Display link*, u kojem su određene početne koordinate svakog signala s kamere i njihova širina i visina na ekranu.

Ovaj korisnički slučaj koristi se na dva načina. U jednom slučaju (sa slike 3.1.) koriste se kamere C2 i C4 (zadnja širokokutna i lijeva uskokutna), a u drugom kamere C2 i C5 (zadnja širokokutna i desna uskokutna). Obzirom na to da su na SC SoC-u kamere uvijek grupirane u parovima u jednom redu, za oba korisnička slučaja potrebno je koristiti dva reda (četiri kamere), iako koristimo samo dvije kamere u određenom trenutku. Kamere su na ADAS Alpha ploču priključene tako da su parovi kamera sljedeći: C1 i C2, C3 i kamera koja se ne koristi, te C4 i C5.

Svaki par kamera predstavlja jedan red, a svaka kamera jedan signal u tom redu. Upravo zbog toga potrebno je adekvatno rasporediti signale s kamera da bi se dobio željeni prikaz. Tako se u *Select linku* određuje da se kamere C1 i kamera koja se ne koristi šalju na pripadajuće *Null linkove*, a kamere C2 i C4 (u slučaju prikaza stražnje i lijeve strane) i kamere C2 i C5 (u slučaju prikaza stražnje i desne strane) šalju na daljnju obradu. Izvršavanje algoritma *Mirror* na dvije kamere raspoređeno je tako da se za jednu kameru algoritam izvršava na DSP1 procesoru, a za drugu na DSP2 procesoru, što je na slici 3.7. i naznačeno različitom bojom). U VPE *linku* određeno je da svaka kamera zauzima polovicu ekrana (lijevo i desno).



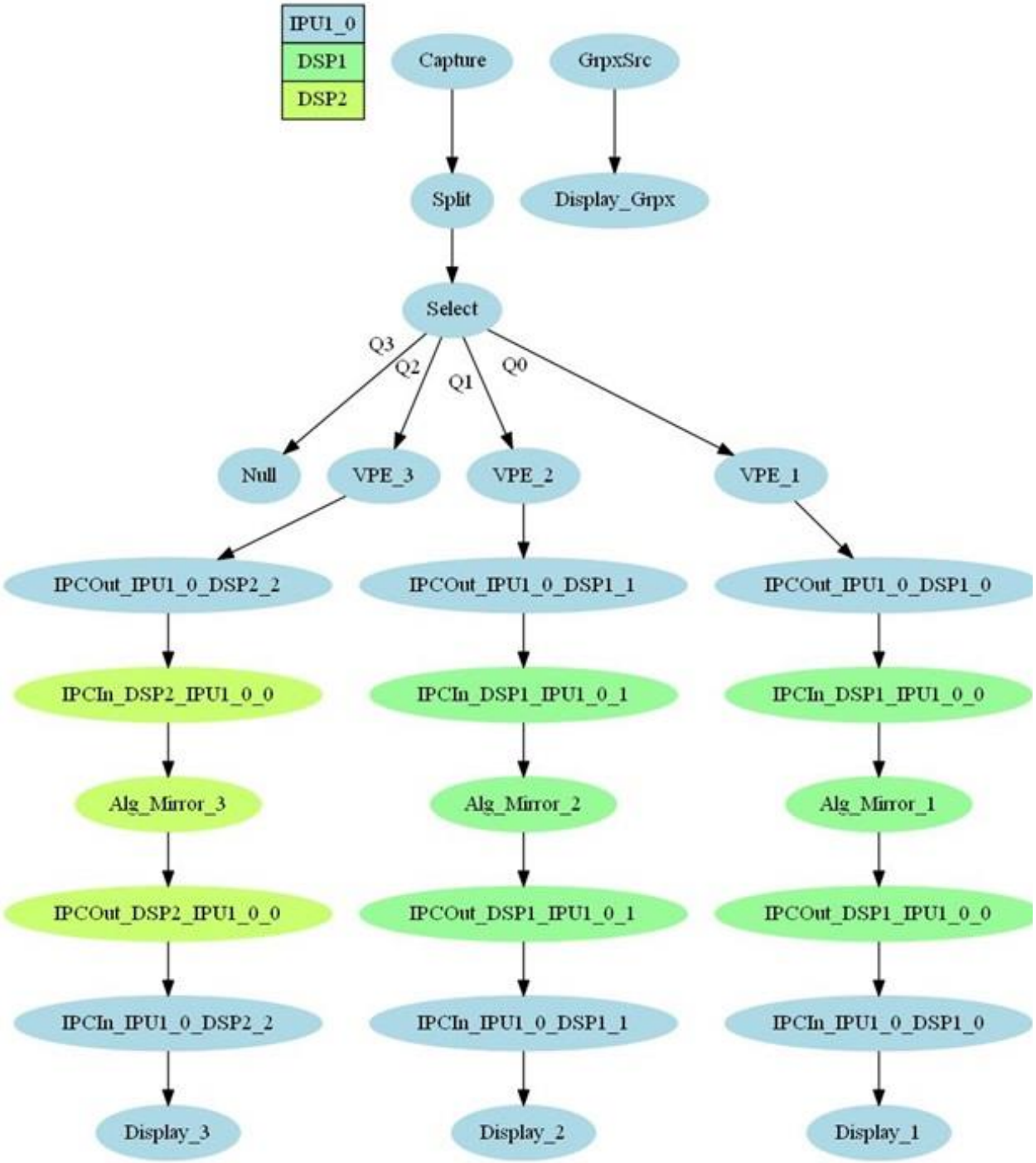
Sl. 3.7. Korisnički slučaj s dvije kamere.

3.4. Korisnički slučaj s tri kamere

U ovom korisničkom slučaju korištene su iste kamere kao u prethodnom potpoglavlju. Rezolucija signala s kamera korištenih u ovom radu je 1280 x 720 elemenata slike, a u izlaznom signalu koristeći *VPE link* povećana je na 1920 x 1080 elemenata slike. Korisnički slučaj s tri kamere u kojem se konstantno izvršava algoritam *Mirror* prikazan je na slici 3.8. Osmišljen je tako da se upotrijebe 3 širokokutne kamere. Razlog tome je pokrivanje što veće površine iza vozila, čime je osiguran pregled oba mrtva kuta. Ovaj korisnički slučaj koristi (sa slike 3.1.) kamere C1, C2 i C3. Obzirom na to da se koriste tri kamere, u ovom slučaju će signal samo jedne nepotrebne kamere biti poslan na *Null link* (jer je zbog ranije navedenog svojstva SC SoC-a potrebno poslati signal sa četiri kamere).

Izvršavanje algoritma *Mirror* nešto je drugačije u ovom korisničkom slučaju. Koriste se tri kamere, a postoje samo dva DSP procesora. Stoga je određeno da dvije kamere budu raspoređene na DSP1 procesor, a jedna na DSP2, što je za ovaj korisnički slučaj optimalna raspodjela za procesiranje signala.

U ovom korisničkom slučaju kamera na zadnjem dijelu vozila pozicionirana je centralno, te je u *VPE linku* odabrano da zauzima pola ekrana. Kamere na lijevoj i desnoj strani vozila raspoređene su lijevo i desno od središnje i svaka zauzima četvrtinu ekrana. Mogućnost izrezivanja dijela slike korisna je zbog nepovoljnog položaja kamere (a i velikog FoV-a širokokutnih kamera C1 i C3) na ADAS Alpha ploči zbog toga što se u originalnom prikazu previše vidi sama maketa vozila (slika 3.3.), a ne okolina oko njega. Empirijskim određivanjem koordinata za izrezivanje dobiven je pregled cijele okoline vozila bez suvišnog prikaza makete vozila.



Sl. 3.8. Korisnički slučaj s tri kamere.

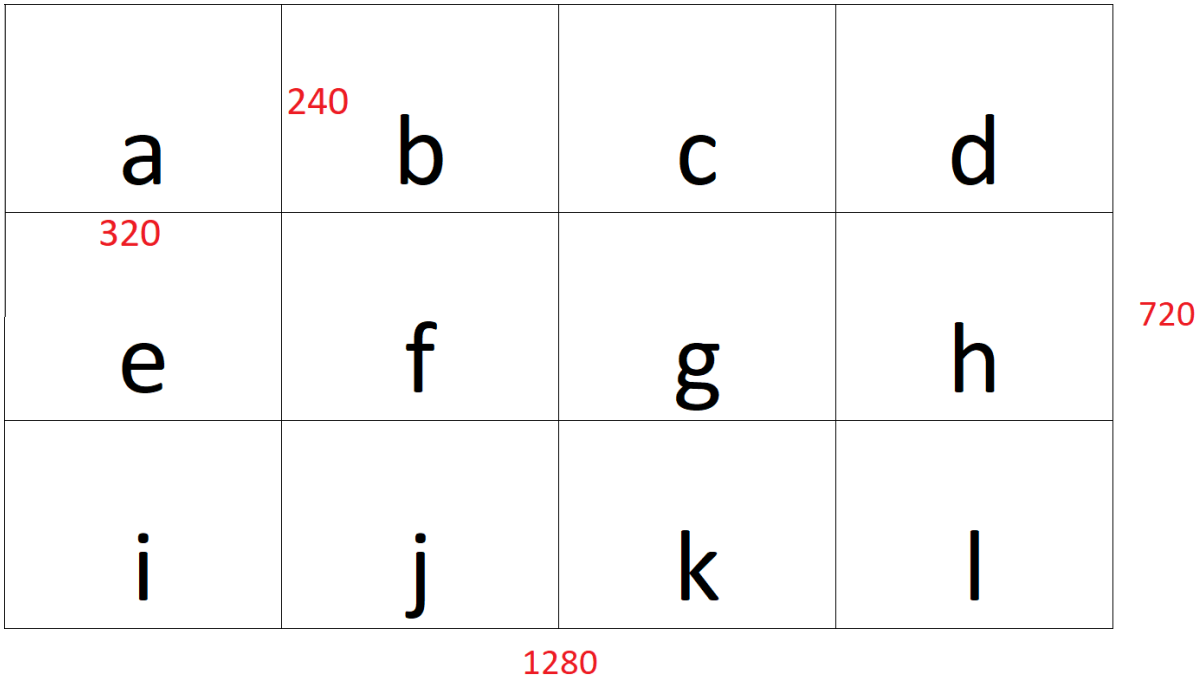
3.5. Korisnički slučaj s jednom kamerom i uvećavanjem dijela slike

Korisnički slučaj s jednom kamerom i uvećavanjem dijela slike osmišljen je uglavnom za pomoć pri parkiranju vozila ili pri vožnji unatrag. Pokreće se na kameri na zadnjoj strani vozila, a izabrana kamera je širokokutna, pa vozač može vidjeti gotovo 180 stupnjeva okoline zadnjeg dijela vozila. Princip rada je uvećavanje proizvoljnog dijela slike. Ovisno o tome koji dio korisnik odabere, taj dio se prikazuje uvećan, a cjelokupnu sliku korisnik može vidjeti u gornjem desnom uglu. Podešena su 2 moguća stupnja uvećavanja. Prvi je uvećanje manjeg dijela slike, a drugi nešto većeg dijela slike.

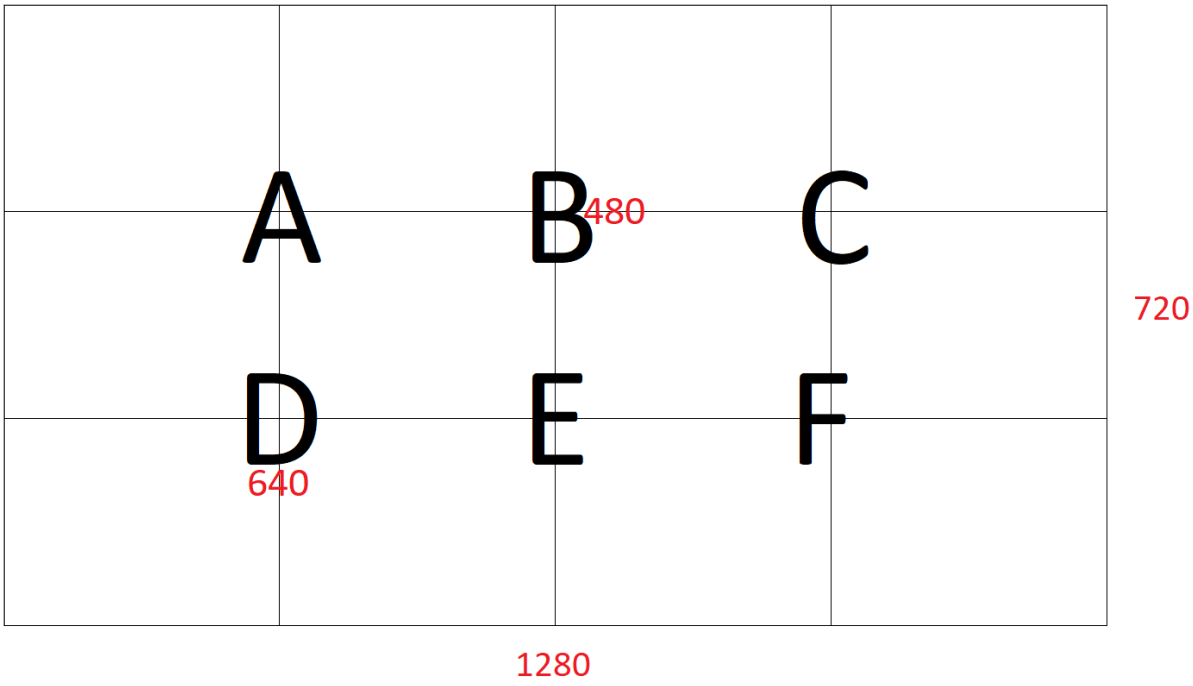
Prvi stupanj uvećanja zasniva se na podjeli širine slike na 4 jednaka dijela, a visine slike na 3 jednaka dijela. Korisnik odabire bilo koji od 12 mogućih dijelova slike za uvećavanje, kako bi mogao vrlo detaljno vidjeti što se nalazi oko vozila. Kako je ulazni signal s kamere u HD rezoluciji (1280x720), dobiveni signal u ovom slučaju ima rezoluciju 320x240 elemenata slike i prikazan je preko cijele izlazne slike na ekranu. Slika 3.9. prikazuje područja prikaza za prvi stupanj uvećanja.

Drugi stupanj uvećanja također koristi podjelu slike po širini na 4, a visine na 3 dijela. Razlika od prvog je u tome što korisnik sada može odabrati veći dio slike za prikaz, u slučaju da mu treba nešto veće područje s manje detalja. U tom slučaju korisnik može uvećati područje veličine polovice širine i dvije trećine visine slike, odnosno 640x480 elemenata slike. Područja prikaza za drugi stupanj uvećanja vide se na slici 3.10. Na slici 3.11. prikazana je ilustracija ekrana za prvi i drugi stupanj uvećanja, a pod opcijama a) i b) navedene su razlike u ta dva stupnja.

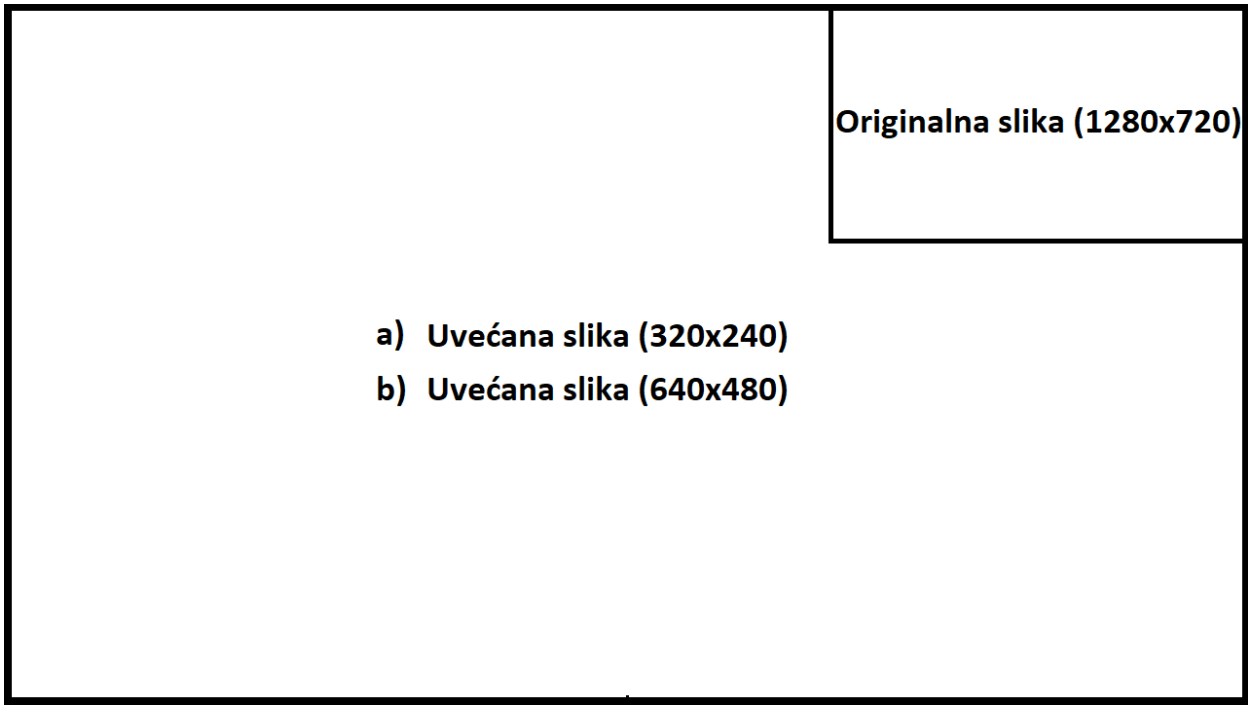
Na slici 3.12. prikazan je izgled korisničkog slučaja s jednom kamerom i uvećanjem dijela slike. U ovom korisničkom slučaju koristi se samo jedan algoritamski *link* (*Mirror*). Operacije tog signala izvršavaju se na DSP1 procesoru. Uvećavanje slike izvršava se u VPE *linku* na način da je za svako područje u slici prethodno definirana početna koordinata i širina i visina isječka. Nakon toga se u *Display linku* određuje pozicija slike na ekranu. Izvorni signal s kamere prikazuje se u gornjem desnom kutu, pri čemu zauzima 1/3 širine i 1/3 visine ekrana. Odabrani dio slike koji se uvećava prikazan je preko ostatka ekrana.



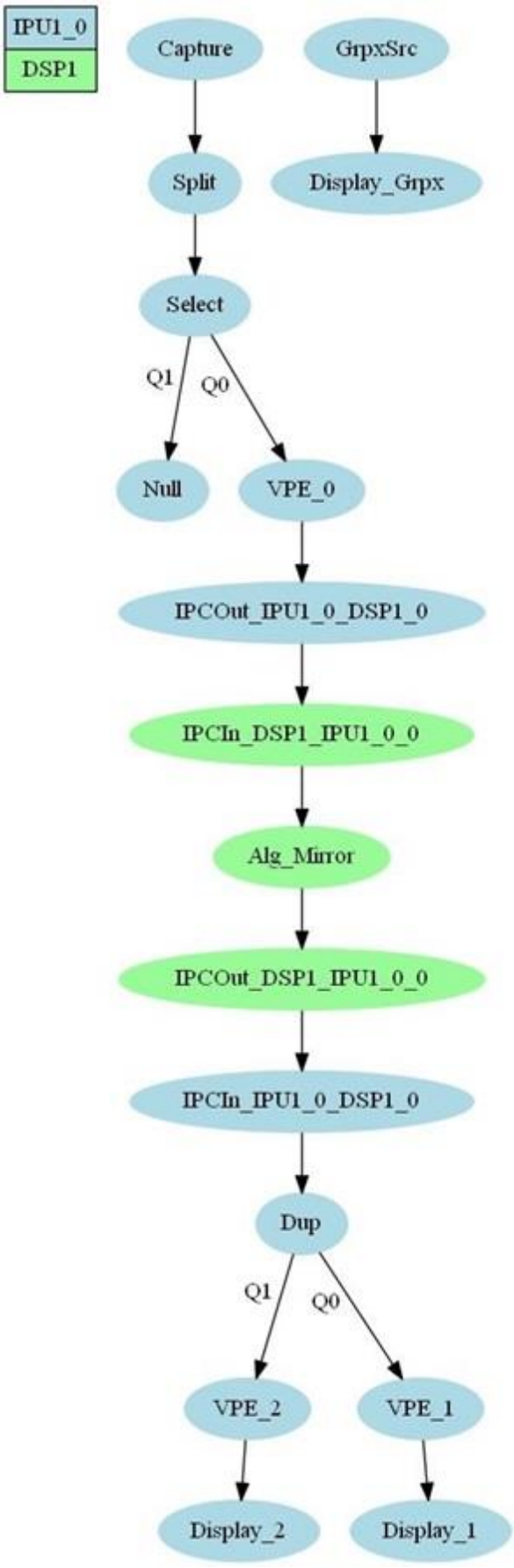
Sl. 3.9. Područja u slici dobivenoj s kamere za prvi stupanj uvećavanja.



Sl. 3.10. Područja u slici dobivenoj s kamere za drugi stupanj uvećavanja.



Sl. 3.11. Raspored prikaza na ekranu za:
a) prvi stupanj uvećanja
b) drugi stupanj uvećanja.



Sl. 3.12. Korisnički slučaj s jednom kamerom i uvećavanjem dijela slike.

3.6. Korisnički slučaj za procjenu udaljenosti i izračunavanje brzine vozila

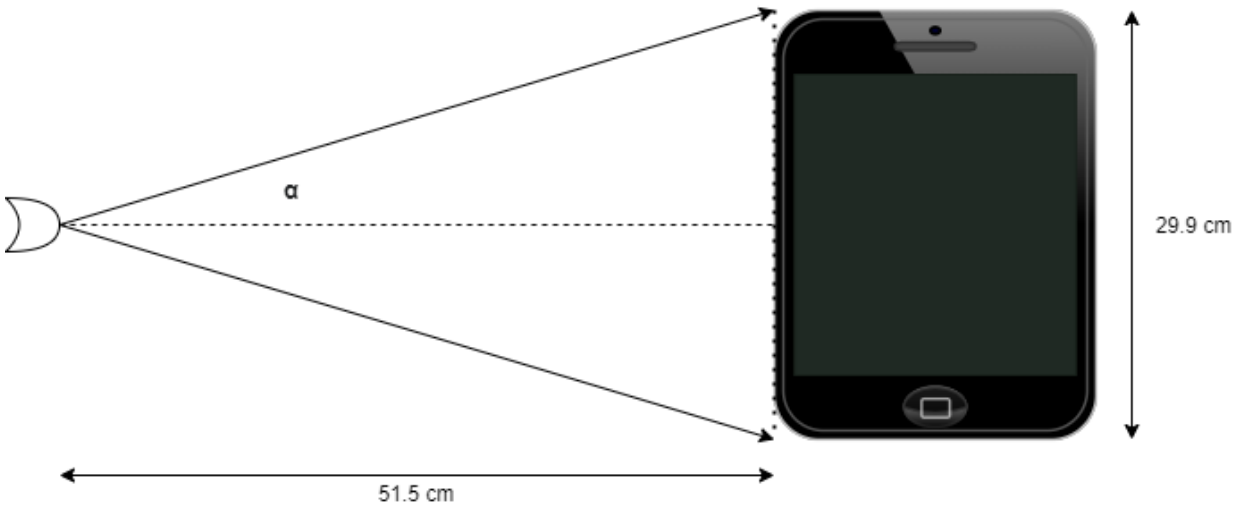
Osim što stražnje kamere mogu služiti za proizvoljni zrcaljeni prikaz scene iza vozila, u ADAS sustavima moguće je implementirati i neke dodatne funkcionalnosti. U ovom potpoglavlju predložen je korisnički slučaj u kojem se vrši detekcija automobila na sceni, pri čemu se procjenjuje brzina kojom se to vozilo kreće s obzirom na vlastito vozilo i upozorava se vozača ako je vozilo koje je iza preblizu ili se kreće prebrzo s obzirom na vlastitu brzinu. Za tu svrhu u VISION SDK programskom okruženju postoji već implementirani algoritam za detekciju objekata, pri čemu korisnik može izabrati detekciju vozila, pješaka ili prometnih znakova, a dodatno je implementirana logika za procjenu udaljenosti i brzine vozila.

Svaki detektirani objekt algoritam za detekciju objekata označava kvadratom koji uokviri objekt (engl. *Bounding Box – BB*). Detektirani pješaci označeni su ljubičastom, znakovi plavom, a vozila zelenom bojom BB-a. Za svaki BB poznate su njegove karakteristike: početne koordinate (vertikalna i horizontalna), širina i visina. Vozila su u algoritmu detektirani tako da kvadrat (svaki BB kvadratnog je oblika) bude raširen točno po visini automobila, a po širini ne. U tom slučaju širina BB-a manja je od širine automobila. Zbog toga što visina BB-a bolje označava automobil, ona se koristi za daljnje računanje.

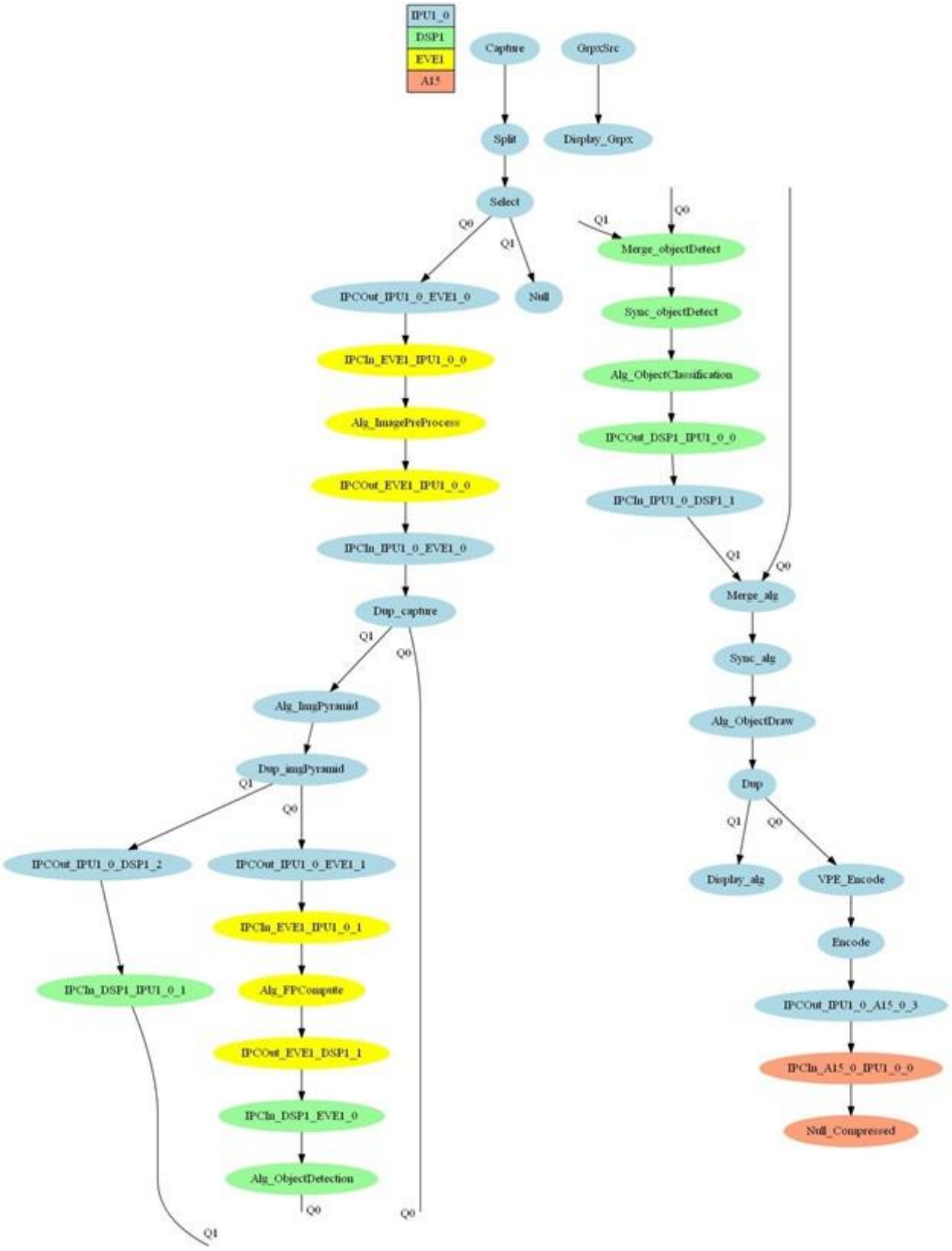
Za potrebe ovog korisničkog slučaja odabrana je kamera C4 sa slike 3.1. Kako se i ovaj korisnički slučaj izvršava na SC SoC-u, u *Capture linku* se šalju dvije kamere u paru, a jedna od njih šalje se na *Null link*. Za potrebe ovog korisničkog slučaja dovoljna je i uskokutna kamera jer je cilj detektirati vozilo koje nailazi u susjednoj traci. Sliku s takve kamere nije potrebno ispravljati zbog neznatne izobličenosti za razliku od slika koje se dobivaju od širokokutnih kamera. Postupak za *Split* i *Select linkove* isti je kao u prethodnim korisničkim slučajevima. Već implementirani algoritam za detekciju objekata sadrži više *linkova*: *ImagePreProcess* (za pretprocesiranje slike), *Dup* (za udvostručavanje ulaznog signala, *ImgPyramid* (za obradu slike), *FPCompute* (za upravljanje memorijskim jedinicama), *ObjectDetection* (za detekciju objekata), *Merge* (za spajanje više redova u jedan), *Sync* (za sinkronizaciju informacijskih tokova), i *ObjectClassification* (za klasifikaciju objekata), *ObjectDraw* (za iscrtavanje željenih grafičkih elemenata u algoritmu) – izlaz iz algoritma *ObjectDraw* je slika s nacrtanim BB-evima. Najprije je dodan dio kôda koji služi za snimanje komprimiranog videa koji je izlaz iz algoritma za detekciju objekata za potrebe razvoja i verifikacije rada algoritma na računalu. Općenito, snimanje videa na ADAS Alpha ploči vrši se pomoću *Null linka* na A15 procesoru. Moguće je preko ethernet veze slati i primati video okvire između ploče i neke druge periferije (u ovom primjeru korišteno

je osobno računao). Ethernet sučelje ne može pružiti dovoljnu propusnost za slanje videa koji nije kompresiran (u izvornom YUV formatu s poduzorkovanjem 4:2:2) – to se manifestiralo tako da nije prebačen svaki video okvir. Zato je u korisnički slučaj prije slanja videa na Null *link* ubačeno kodiranje videa prema MJPEG normi. Snimljeni video signal ima 15 FPS-a video okvira po sekundi (engl. *Frames Per Second – FPS*) i na njemu je prikazan signal s kamere sa BB-evima detektiranih vozila. Naposljetku je dodan kôd za procjenu udaljenosti i procjenu brzine automobila u algoritamski *link ObjectDraw* koji omogućava signaliziranje vozaču ukoliko je detektirani automobil bliže od postavljene sigurnosne udaljenosti ili se kreće brže od postavljene sigurnosne brzine kretanja.

Prije računanja brzine detektiranog automobila potrebno je odrediti njegovu udaljenost od kamere u svakom video okviru signala. Za to je prvo potrebno odrediti vertikalni kut kamere (zbog toga što će se izračun temeljiti na visini BB-a detektiranog vozila). Kut je određen postavljanjem scene sa slike 3.13. u kojoj je izmjerena udaljenost objekta do kamere i visina objekta. Fiksno je postavljen objekt poznate visine tako da zauzme točno cijelo područje kamere po visini, a to je utvrđeno na zaslonu monitora na kojem je bio signal s kamere. Slika korisničkog slučaja za procjenu udaljenosti i izračunavanje brzine vozila prikazana je na slici 3.14.



Sl. 3.13. Izgled scene za računanje kuta kamere.



Sl. 3.14. Korisnički slučaj za detekciju vozila i snimanje video signala koji sadrži ulazni signal s video kamere i detektirane BB-ove.

Izraz za izračun kuta kamere zadan je izrazom (3-1), a rezultat u stupnjevima i radijanima nalazi se u izrazima (3-2) i (3-3).

$$tg(\alpha) = \frac{14.95}{51.5} \tag{3-1}$$

$$2\alpha = 32.375° \tag{3-2}$$

$$2\alpha = 0.5650498084\, rad \tag{3-3}$$

Rezolucija slike dobivene s kamere 1280x720 elemenata slike, pa vrijedi omjer (3-4) iz kojeg je dobiven podatak koliki kut kamere odgovara jednom elementu slike, tj. njegovoj visini. Iznos te konstante izražen je u izrazu (3-5) .

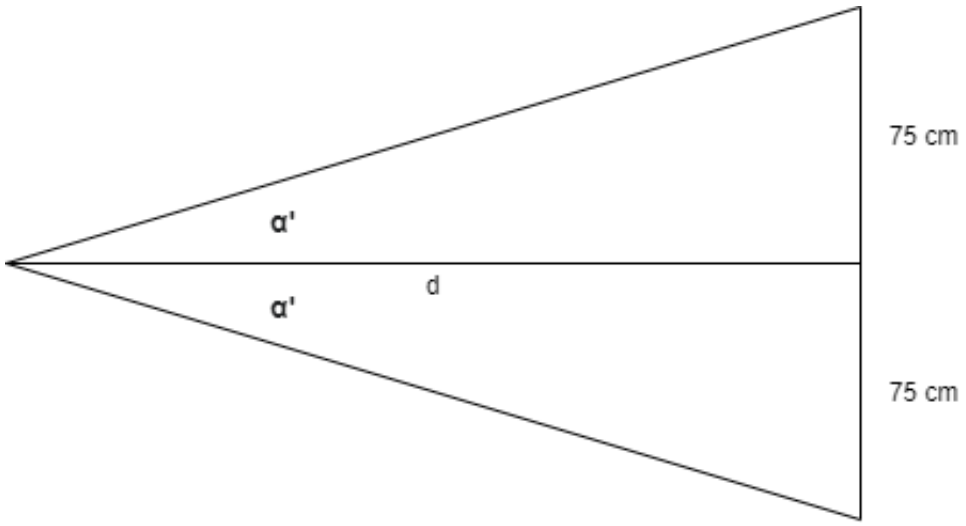
$$2\alpha : 720 = x : 1 \tag{3-4}$$

$$x = 0.0007847914\ \frac{rad}{el.slike} \tag{3-5}$$

Poznavajući ovisnost kuta kamere o visini objekata u slici ili cijele slike, može se jednostavno odrediti kut kojeg zatvara bilo koji objekt kojem je poznata visina u elementima slike. Tako se za svaki novodetektirani objekt prvo računa kut kojeg visina BB-a zatvara s kamerom (jer visina BB-a najbolje određuje i samu visinu objekta) tako da se prema (3-6) pomnože visina trenutnog BB-a i konstanta x.

$$2\alpha' = BB_{visina} * x \tag{3-6}$$

Da bi se dobio konačni podatak o udaljenosti automobila od kamere, potrebno je odrediti prosječnu visinu realnog automobila, tj. visinu koja će predstavljati svaki automobil. Ona iznosi između 1.5 i 1.8 metara [12]. U ovom radu izabrana je visina od 1.5 metara. Prema tome izvode se nova geometrijska postava (slika 3.15.) i izračun za udaljenost automobila od kamere (3-7).



Sl. 3.15. Geometrijska postava za računanje kuta kamere kojeg zatvara detektirani objekt sa kamerom.

$$d = 0.75 * tg(\alpha') \tag{3-7}$$

Za svaki detektirani objekt se najprije odredi kut $2\alpha'$ prema izrazu (3-6). Zatim se procijeni udaljenost vozila d pomoću izraza (3-7). Na temelju procjene udaljenosti detektiranog vozila za svaki video okvir moguće je procijeniti brzinu kretanja detektiranog vozila. Izračunavanje udaljenosti d detektiranog vozila izvršava se pri svakoj detekciji (svakom video okviru), pa se može dobiti promjena puta između dvije detekcije, odnosno dva dobivena video okvira. Formula za brzinu detektiranog vozila $v_d(k * T_s)$ u određenom trenutku dana je izrazom (3-8), pri čemu je v_0 brzina vlastitog vozila (koja je jednaka 0 u okviru ovog rada jer je maketa vozila mirovala tijekom testiranja, pa se brzina detektiranog vozila $v_d(k * T_s)$ može dobiti i izrazom (3-9)). Varijabla k označava redni broj video okvira, a T_s vrijeme između dva video okvira. Brzina detektiranog vozila jednaka je promjeni udaljenosti između dvije detekcije Δs u nekom periodu vremena (3-10). Promjena puta definirana je izrazom (3-11).

$$v_d(k * T_s) = v_0 + \Delta v(k * T_s) \tag{3-8}$$

$$v_d(k * T_s) = \Delta v(k * T_s) \tag{3-9}$$

$$\Delta v(k * T_s) = \frac{\Delta s}{T_s} \tag{3-10}$$

$$\Delta s = d(k * T_s) - d((k-1) * T_s) \tag{3-11}$$

Jedina nepoznanica koja preostaje je vrijeme trajanja jednog video okvira T_s. Ta se konstanta može dobiti pomoću funkcije *Utils_getCurTimeInMsec()*. Funkcija kao povratnu vrijednost vraća vrijeme potrebno da korisnički slučaj obradi jedan video okvir. To vrijeme iznosi 0.0667 sekundi. Na temelju ove vrijednosti moguće je izračunati i broj okvira koji se uspiju obraditi u sekundi (FPS). Ta vrijednost iznosi 15. Prema postupku na slici 3.16. može se odrediti brzina vozila iza vlastitog vozila. Postavljena su 2 brojača (*currFrame* i *refFrame*) koji su postavljeni tako da su uvijek razmaknuti za jedan video okvir. Algoritam u liniji 2 u prvom video okviru spremi trenutnu procijenjenu udaljenost. Za uspješnost računanja udaljenosti potrebna je razlika dvaju brojača u iznosu od jedan. Kako bi se to osiguralo, ukoliko se dogodi da *currFrame* bude veći za više od jedan, vrijednosti brojača se izjednačavaju, što osigurava daljnju mogućnost ispravnog rada, a to je navedeno u linijama 5,6 i 7. Ako je uvjetno grananje u liniji 10 ispunjeno, to znači da je proteklo vrijeme jednako 1/15 sekunde. Sprema se procjena udaljenosti za trenutni video okvir i u liniji 12 računa se razlika te udaljenosti i udaljenosti spremljene video okvir ranije. Upravo ta razlika predstavlja put prijeđen u 1/15 sekunde, odnosno brzinu u metrima po 1/15 sekunde. Nakon toga se za referentni brojač postavlja trenutni i za referentnu udaljenost trenutna udaljenost. U liniji 15 izračunava se brzina u km/h. Dobiveni rezultat određuje vrijednost za koju se promijenila brzina detektiranog vozila, odnosno koliko je detektirani automobil sporiji ili brži od automobila na kojem se nalazi kamera. U ovom radu kamera miruje, pa je zato ta vrijednost ujedno i apsolutna brzina kretnja detektiranog automobila. Od linije 18 do 22 postavljeni su uvjeti obavještavanja vozača za sporiju i bržu vožnju drugog automobila. U liniji 25 uvećava se marker *currFrame* za jedan.

Linija Kod

```
1:      if( currFrame == 0 ){
2:          refDistance = pObjectDataDesc-> reserved0;
3:      }
4:
5:      if (currFrame > refFrame+1 ){
6:          refFrame = currFrame;
7:          continue;
8:      }
9:
10:     if( (currFrame-1) == refFrame){
11:         currDistance = pObjectDataDesc-> reserved0;
12:         difference = refDistance - currDistance;
13:         refFrame = currFrame;
14:         refDistance = currDistance;
15:         speed=difference*54;
16:      }
17:
18:     if(speed > 0){
19:         Vps_printf("Automobil je brzi %.2f km/h\n",(speed));
20:     }
21:     else{
22:         Vps_printf("Automobil je sporiji %.2f km/h\n",(-1)*(speed));
23:     }
24:
25:     currFrame++;
26:
27:
```

Sl. 3.16. Kôd algoritma za određivanje brzine detektiranog vozila.

Udaljenost se sprema u varijablu *pObjectDataDesc-> reserved0* koja je dio strukture korisničkog slučaja. Brzina u km/h spremljena je u varijablu *speed*. Pri detekciji automobila u VISION SDK programskom okruženju preddefinirana boja BB-a je zelena. Kao način upozoravanja vozača u ovom radu osmišljena je promjena boje BB-a u crvenu i ispisivanje odgovarajuće poruke na ekran. Implementacija navedenog signaliziranja vidljiva je na slici 3.17. Prema postavljenim uvjetima u linijama 1 i 2, sustav će vozača upozoriti ako je :

- Detektirano vozilo brže za 5 km/h i uz to bliže od 10m od vozila
- Detektirano vozilo bliže od 5 m od vozila neovisno o brzini
- Detektirano vozilo brže za 15 km/h neovisno o blizini

Linija Kod

```
1:      if(((speed > 5) && (pObjectDataDesc-> reserved0 < 10)) ||
2:          (pObjectDataDesc-> reserved0 < 5) || (speed>15))
3:      {
4:                  pObj->linePrm.lineColor = COLOR_RED;
5:                  Vps_printf("***** NIJE SIGURNO PRESTROJITI SE! *****\n");
6:      }
7:      else{
8:                  pObj->linePrm.lineColor = COLOR_GREEN;
9:      }
```

Sl. 3.17. Kôd na temelju kojeg se upozorava vozača.

4. VERIFIKACIJA RADA ALGORITAMA ZA KAMERE KAO ZAMJENU ZA ZRCALO U VOZILIMA

Verifikacija predloženih algoritama iz trećeg poglavlja izvršena je praćenjem rezultata preko HDMI izlaza na ekranu računalnog monitora i u aplikaciji Tera Term koja služi za simulaciju različitih računalnih terminala. Podržava serijsku komunikaciju, a SoC ugrađen na ADAS Alpha ploču također ima UART sučelje. Pomoću ispisa u aplikaciji Tera Term, određena je optimiziranost raspodjele računalnih zadataka po procesorima, efikasnost detekcije vozila i procjena udaljenosti detektiranog vozila od makete vozila. Testiranje je provedeno uz pomoć osobnog računala na kojem je pokrenuta aplikacija Tera Term i na kojem je implementirano VISION SDK programsko okruženje.

4.1. Korisnički slučaj s dvije kamere – verifikacija rada

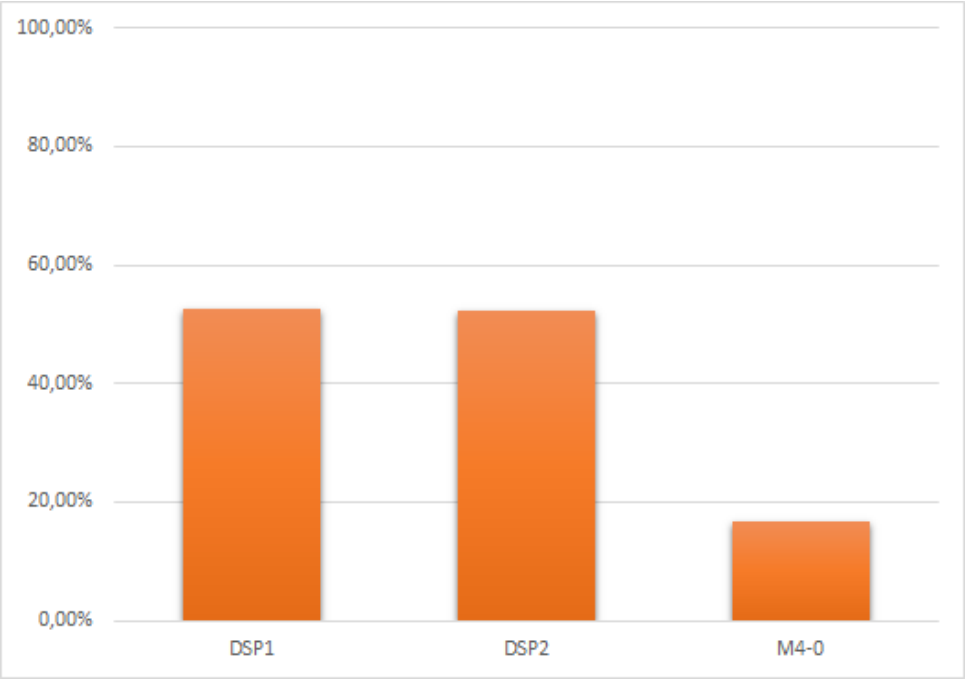
Korisnički slučajevi pokrenuti na ADAS Alpha ploči koriste algoritam za zrcaljenje slike (*Mirror*). Slika je točno zrcaljena i nema vidljivih grešaka ili neželjenih pojava pri korištenju algoritma. Kako bi se izvršila verifikacija ovog korisničkog slučaja u kôd algoritamskog *linka Mirror* ubačena je funkcija *Utils_getCurTimeInMsec()* koja kao povratnu vrijednost vraća vremensku oznaku. Pomoću funkcije *Vps_printf() za* svaki se video okvir ispisuje vremenska oznaka pomoću koje se može odrediti FPS koji je recipročna vrijednost navedenoj vremenskoj oznaci. Na slikama 4.1. i 4.2. nalaze se izlazni signali s kamera na ekranu monitora. Korisnički slučaj je pokrenut tako da traje minimalno dvije minute te je izračunata srednja vrijednost FPS-a koja je iznosila 29.9. Pri tome je najmanja vrijednost FPS-a iznosila 29. Postignuta vrijednost FPS-a dovoljna je za ovaj korisnički slučaj. Korisnik subjektivno ne može primijetiti razliku pri minimalnoj vrijednosti. FPS korisničkog slučaja odgovara FPS-u kamere, a iznosi 30. Algoritam *Mirror* ne uzrokuje gubitak video okvira.

Također je izmjereno opterećenje svih procesora koji sudjeluju u pokretanju ovog korisničkog slučaja. Korisnički slučaj raspoređen je na dva DSP procesora (na njima je algoritam radio optimalno – na EVE i A15 procesorima empirijski je određeno da radi nezadovoljavajuće, tj. korisnički slučaj je preskakao video okvire). Kako postoje samo dva DSP procesora, a u ovom korisničkom slučaju koriste se dvije kamere čiji se signali trebaju obraditi, simetrično je obrada raspoređena po jedna na svaki od dva DSP procesora. Procesor M4 koristi se za dohvaćanje video okvira, kodiranje i dekodiranje signala, udvostručavanje signala, prikazivanje signala na ekranu i

sl. Rezultati opterećenja procesora odgovaraju tom odnosu, a prikazani su na slici 4.3. Iz slike se vidi da je na svim procesorima moguće implementirati dodatno procesiranje.

Sl. 4.1. Prikaz izlaznog signala na ekranu za korisnički slučaj s dvije kamere – zadnja i lijeva kamera

Sl. 4.2. Prikaz izlaznog signala na ekranu za korisnički slučaj s dvije kamere – zadnja i desna kamera



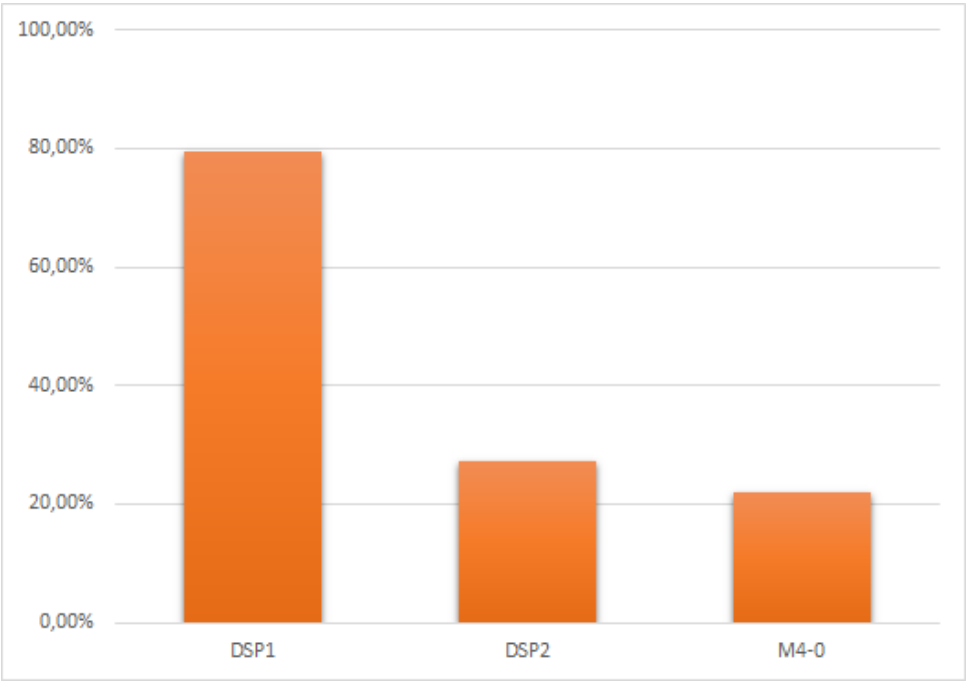
Sl. 4.3. Opterećenja procesora za korisnički slučaj s dvije kamere.

4.2. Korisnički slučaj s tri kamere – verifikacija rada

U ovom korisničkom slučaju korištena je ista metoda verifikacije kao u potpoglavlju 4.1. Na slici 4.4. prikazan je izlazni signal s kamera na ekranu monitora. Korisnički slučaj je pokrenut tako da traje minimalno dvije minute, a zatim je izračunata srednja vrijednost FPS-a koja je također iznosila 29.9. Pri tome je najmanja vrijednost FPS-a iznosila 29. Postignuta vrijednost FPS-a dovoljna je i za ovaj korisnički slučaj - korisnik subjektivno ne može primijetiti razliku pri minimalnoj vrijednosti. FPS korisničkog slučaja odgovara FPS-u kamere, a iznosi 30. Algoritam *Mirror* ne uzrokuje gubitak video okvira.

S druge strane, opterećenje procesora je nešto drugačije nego u potpoglavlju 4.1. Razlog tome je korištenje tri kamere, a samo su raspoloživa dva DSP procesora na TDA2xx SoC-u. Tako su tri signala s kamera podijeljena na dva DSP procesora na način da su dva signala na procesoru DSP1, a jedan preostali signal na procesoru DSP2. Rezultati opterećenja procesora odgovaraju tom odnosu, a prikazani su na slici 4.5. - iz slike se vidi da je moguće implementirati dodatno procesiranje, a pogotovo na procesoru DSP2.

Sl. 4.4. Prikaz izlaznog signala na ekranu za korisnički slučaj s tri kamere



Sl. 4.5. Opterećenja procesora za korisnički slučaj s tri kamere.

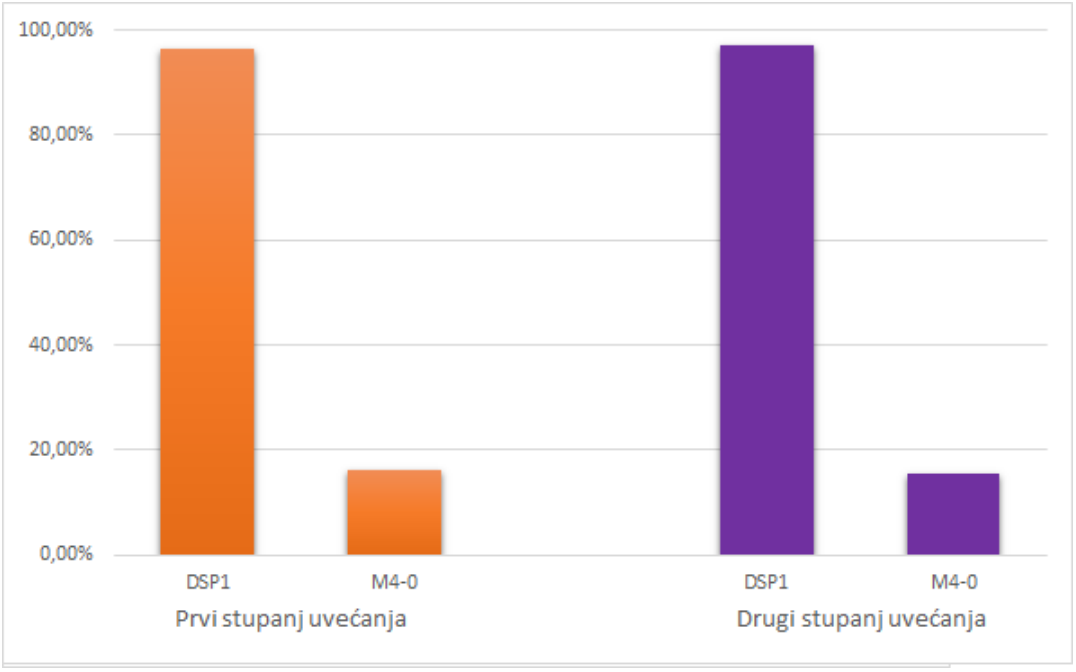
4.3. Korisnički slučaj s jednom kamerom i uvećavanjem dijela slike – verifikacija rada

U ovom korisničkom slučaju također je korištena ista metoda verifikacije kao u prethodnim potpoglavljima. Na slikama 4.6. i 4.7. prikazani su izlazni signali s kamera na ekranu. Korisnički slučaj pokrenut je tako da traje minimalno dvije minute, a zatim je izračunata srednja vrijednost FPS-a koja je također iznosila 29.9. Pri tome je najmanja vrijednost FPS-a iznosila 29. Postignute vrijednosti FPS-a dovoljne su i za ovaj korisnički slučaj - korisnik subjektivno ne može primijetiti razliku pri minimalnoj vrijednosti. FPS korisničkog slučaja odgovara FPS-u kamere, a iznosi 30. Algoritam *Mirror* ne uzrokuje gubitak video okvira ni u ovom korisničkom slučaju.

Opterećenje procesora neznatno se razlikuje za uvećavanje prvog i drugog stupnja. Usporedba opterećenja prikazana je na slici 4.8.

Sl. 4.6. Prikaz izlaznog signala na ekranu za korisnički slučaj s uvećavanjem dijela slike – prvi stupanj uvećanja

Sl. 4.7. Prikaz izlaznog signala na ekranu za korisnički slučaj s uvećavanjem dijela slike – drugi stupanj uvećanja



Sl. 4.8. Opterećenja procesora za korisnički slučaj s uvećavanjem dijela slike – usporedba prvog i drugog stupnja uvećanja.

4.4. Korisnički slučaj za procjenu udaljenosti i procjenu brzine vozila – verifikacija rada

Za verifikaciju ovog korisničkog slučaja korišteno je pet osobnih automobila, a lokacija testiranja je ulica Cara Hadrijana 10b u Osijeku tijekom sunčanog dana. Korišteni automobili su: Renault Megane 2004 (V1), Audi A4 2005 (V2), Citroen C4 2006 (V3), Audi A3 2007 (V4) i Golf VII 2014 (V5). Na scenu su postavljene oznake na svakih pet metara koje služe kao referentne točke za dobivene rezultate. Osvjetljenje je tijekom testiranja bilo promjenjivo. Testni automobili kretali su se prema kameri koja je bila montirana na maketu automobila.

U ovom potpoglavlju prikazani su rezultati testiranja računanja procijenjene udaljenosti za svih pet automobila. U tablici 4.1. prikazane su izmjerene srednje vrijednosti procijenjenih udaljenosti detektiranih vozila. Prilikom testiranja svaki automobil je zaustavljen tako da je prednji kraj automobila u ravnini sa unaprijed određenom referentnom točkom za udaljenost. Srednje vrijednosti izračunate su na temelju 20 mjerenja temeljem visine detektiranog BB-a (vidi 3.6. za detalje).

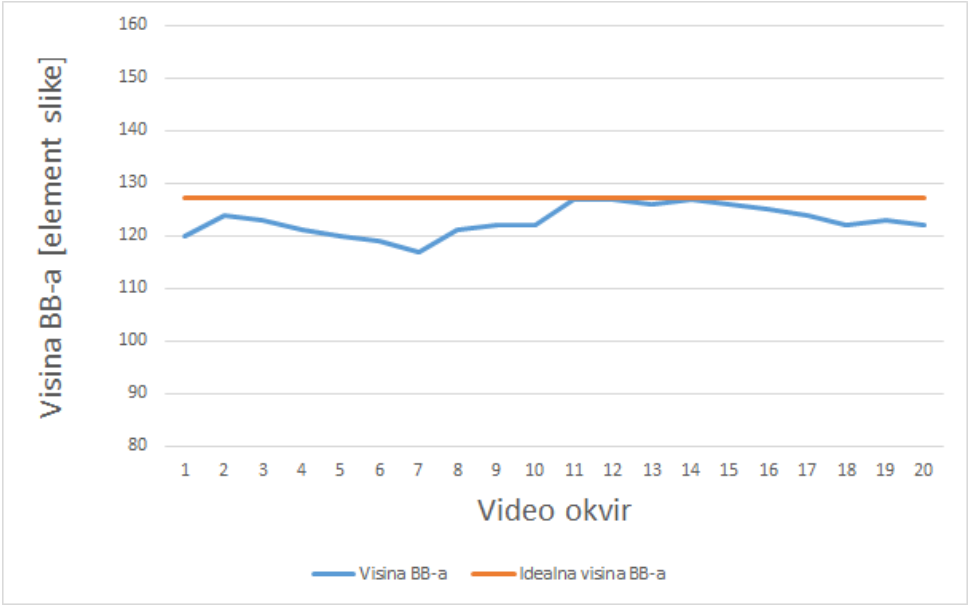
Tablica 4.1. Tablica srednjih vrijednosti procijenjenih udaljenosti detektiranih vozila.

		Stvarna udaljenost [m]					
	Automobil	5	10	15	20	25	30
Srednja vrijednost [m]	V1	6.65	11.95	18.81	28.23	32.70	40.64
	V2	8.07	12.60	16.19	41.65	32.53	45.52
	V3	7.13	11.77	16.28	26.99	28.90	38.11
	V4	nije detektiran	12.16	15.55	28.52	27.15	33.86
	V5	7.85	11.95	19.38	26.68	29.87	38.24

Iz tablice 4.1. se može vidjeti da automobil V4 uopće nije detektiran na udaljenosti 5 m od kamere. Razlozi tome mogu biti bijela boja automobila ili promjena osvjetljenja. Sve ostale vrijednosti veće su od očekivanih – zbog odabrane prosječne visine automobila od 1.5 m, što je nešto više od visine svakog od pet testiranih automobila. Dodatni razlog izračunatih većih udaljenosti je taj što je visina BB-a dobivenog detekcijom pomoću ugrađenog algoritma u gotovo svim slučajevima manja od visine automobila (što implicira da se automobil nalazi dalje nego što

zapravo jest). Rezultati se empirijski mogu popraviti izmjenom prosječne visine automobila, uzevši u obzir statistiku ponašanja algoritma. Primjer dobro detektiranog automobila nalazi se na slici 4.9. – automobil V4 na udaljenosti od 15 m od kamere. Prikaz visine BB-a tijekom uzastopnih video okvira u usporedbi sa idealnom visinom BB-a s obzirom na visinu tog automobila nalazi se na slici 4.10. Može se zaključiti kako je BB uvijek približno jednak očekivanom i prema slici 4.1. vidi se da je srednja pogreška procjene udaljenosti jednaka 0,55 m.

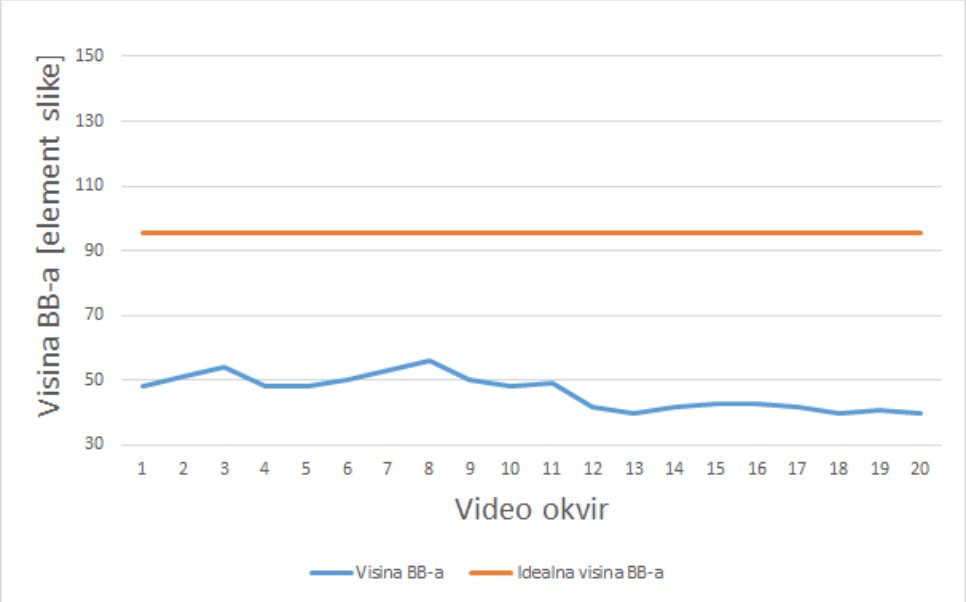
Sl. 4.9. Primjer ispravne detekcije automobila (V4 na udaljenosti 15 m).



Sl. 4.10. Odnos idealne i realne visine BB-a za automobil V4 na udaljenosti 15 m za 20 uzastopnih video okvira.

Na slici 4.11. nalazi se primjer loše detektiranog automobila – automobil V2 na udaljenosti od 20 m od kamere. Prikaz visine BB-a tijekom uzastopnih video okvira u usporedbi sa idealnom visinom BB-a s obzirom na visinu tog automobila nalazi se na slici 4.12. Iz ovih slika se vidi kako je ugrađeni detektor loše lokalizirao automobil, što rezultira nedovoljnom visinom BB-a, a samim time i lošim rezultatima procjene udaljenosti što je vidljivo i u tablici 4.1.

Sl. 4.11. Primjer loše detekcije automobila (V2 na udaljenosti 20 m).



Sl. 4.12. Odnos idealne i realne visine BB-a za automobil V2 na udaljenosti 20 m za 20 uzastopnih video okvira.

Na temelju provedenih mjerenja u tablici 4.2. prikazane su srednja vrijednost apsolutnih odstupanja i maksimalna vrijednost apsolutnih ovisno o tipu testnog automobila, a na u tablici 4.3. prikazani su isti parametri procjene udaljenosti ovisno o referentnim udaljenostima – od 5 m do 30 m u koracima od 5 m.

Tablica 4.2. Srednja i maksimalna vrijednost apsolutnih odstupanja procjene udaljenosti ovisno o tipu testnog automobila.

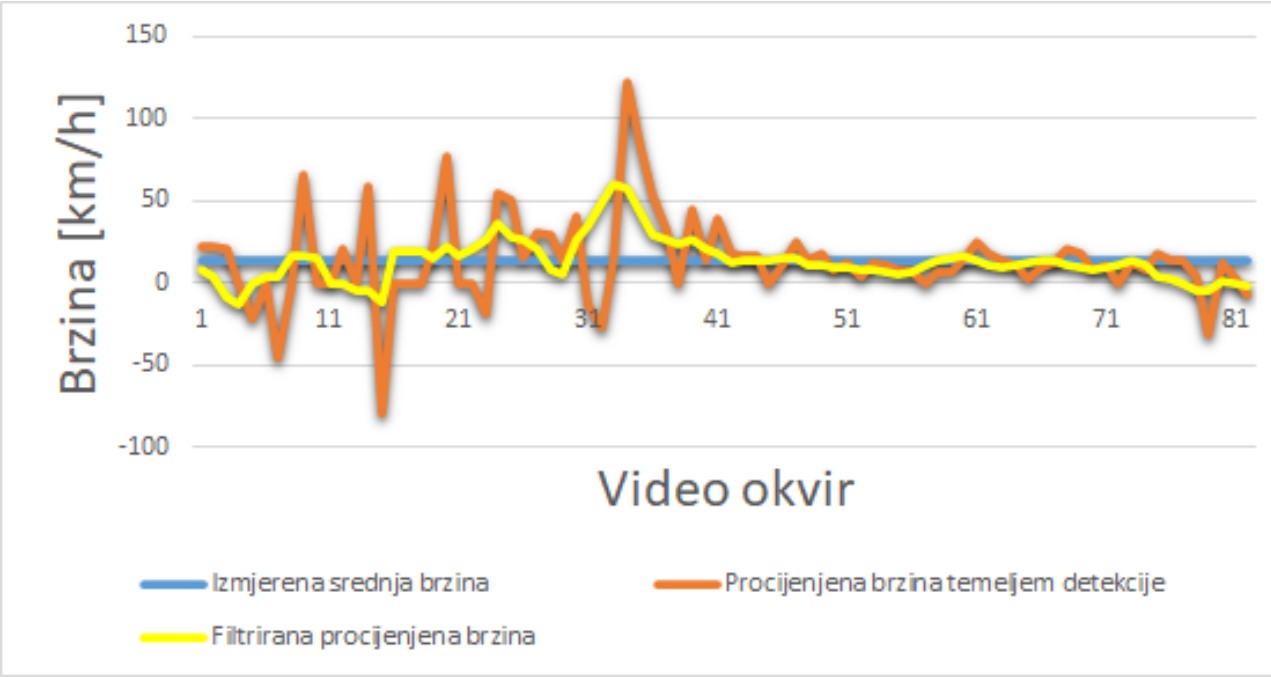
[m]	V1	V2	V3	V4	V5
Srednja vrijednost apsolutnog odstupanja	4.75	7.57	3.21	3.34	4.14
Maksimalno apsolutno odstupanje	10.37	10.82	4.77	4.55	6.54

Tablica 4.3. Srednja i maksimalna vrijednost apsolutnih odstupanja procjene udaljenosti ovisno o referentnim udaljenostima.

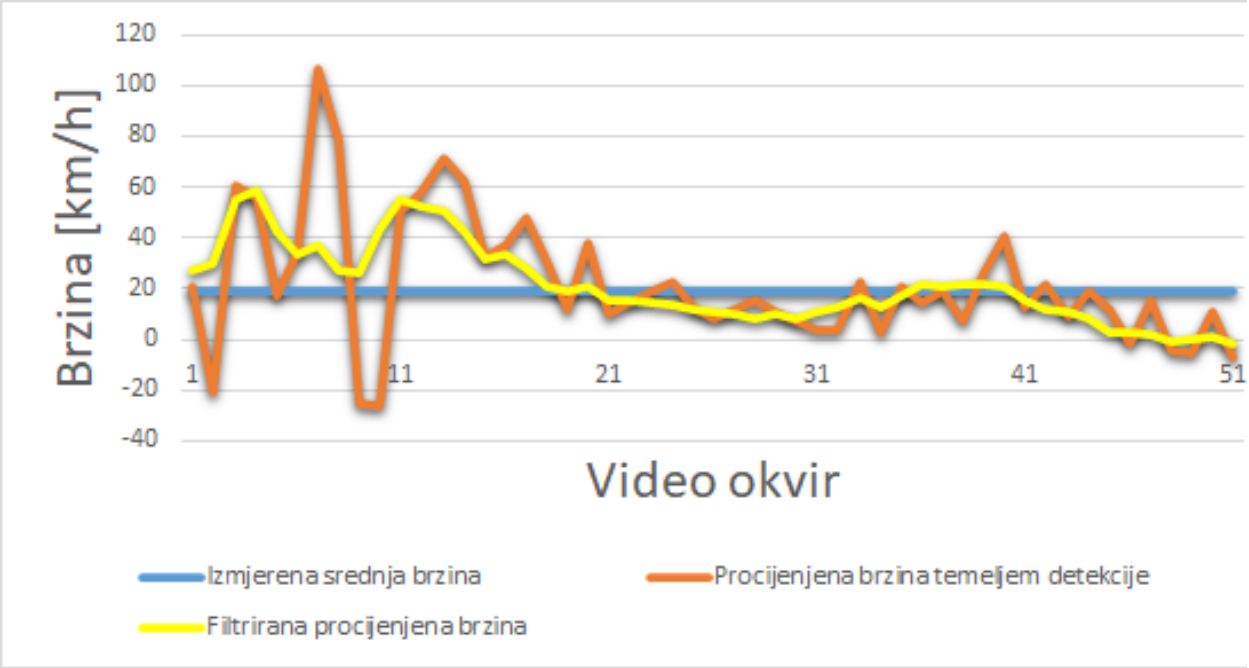
[m]	5m	10m	15m	20m	25m	30m
Srednja vrijednost apsolutnog odstupanja	2.43	2.09	2.60	10.50	5.23	9.27
Maksimalno apsolutno odstupanje	2.83	2.87	3.87	16.52	7.07	10.95

Na temelju mjerenja iz tablica 4.1., 4.2. i 4.3. može se zaključiti kako je automobil V2 nepovoljan za algoritam, što je vidljivo i na slici 4.11. Većinom je algoritam umjesto automobila prepoznavao prednju masku kao automobil – iz tog razloga su rezultati znatno lošiji od rezultata postignutih za druge automobile. Maksimalna odstupanja u oba slučaja rezultat su osciliranja visine BB-a kroz video okvire. Odstupanja su najuočljivija kod svih automobila na udaljenosti od 20 metara. Tome je uvelike doprinijela značajna promjena osvjeljenja scene tijekom izvođenja testiranja. Također se može primijetiti da su maksimalna odstupanja veća su kod većih udaljenosti nego kod manjih.

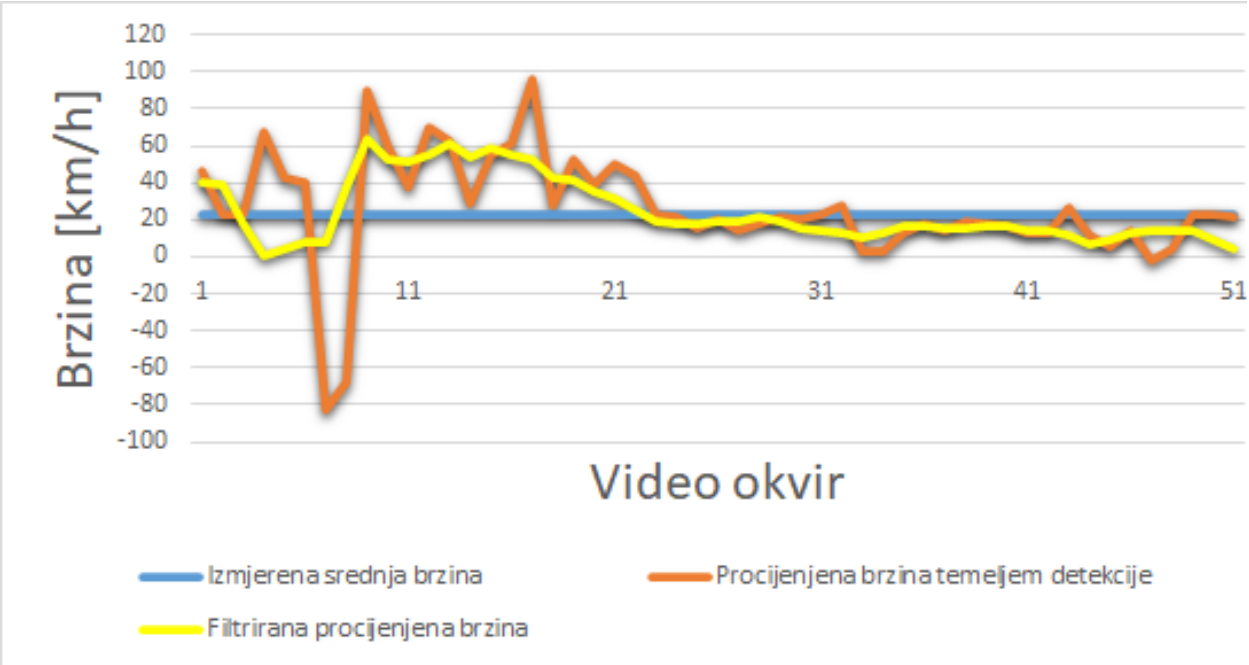
Nadalje, prikazani su rezultati procjene brzine testnog automobila. Mjerenja procjene brzine automobila izvršena su za automobile V3 i V4 jer se za njih pokazalo da detektor najbolje radi, odnosno postignuta je najmanja srednja pogreška odstupanja procjene udaljenosti (vidi tablicu 4.2.). Brzina je procijenjena između svaka dva uzastopna video okvira (što odgovara vremenskom intervalu od 0.067 s). Na slikama 4.13., 4.14. i 4.15. prikazane su ovisnosti izmjerene srednje brzine automobila, procijenjene srednje brzine automobila V3 na temelju procesiranja dobivene detekcije u svakom video okviru i filtrirane procijenjene brzine. Filtriranje je učinjeno filtrom s pomičnim usrednjavanjem veličine pet video okvira, što odgovara vremenu od 0.33 sekunde (engl. *Moving Average Filter*). Srednja brzina mjerena je uz pomoć dvoje studenata koji su stajali na udaljenosti od 20 m i zapisivali vremenske oznake prolaska automobila. Na temelju prijeđenog puta od 20 m i izmjerenog vremena prolaska izračunata je srednja vrijednost brzine (3-10).



Sl. 4.13. Ovisnost srednje i procijenjene brzine vozila V3 pri brzini od 13.26 km/h – korekcija pomoću *moving average* filtra.



Sl. 4.14. Ovisnost srednje i procijenjene brzine vozila V3 pri brzini od 18.65 km/h – korekcija pomoću *moving average* filtra.



Sl. 4.15. Ovisnost srednje i procijenjene brzine vozila V3 pri brzini od 22.43 km/h – korekcija pomoću *moving average* filtra.

Iz prethodnih slika može se zaključiti kako u prvom dijelu vožnje izračunata brzina vožnje više oscilira. Razlog tome je taj što vozač nije držao papučicu gasa tako da se kreće jednolikom brzinom. Također, dodatne oscilacije uzrokuju naglije promjene scene i blizine automobila za razliku od scene kada automobil miruje. Dakle, uz grešku algoritma, pri testiranju postoje greške mjerenja vremena (studenti) i greška vozača (zbog promjenjivog stiskanja papučice za ubrzavanje na automobilu).

Zbog svih navedenih pogrešaka za očekivati je da će raspršenje dobivenih rezultata brzine od očekivanih biti velika. To očekivanje potvrđeno je u tablicama 4.4. za automobil V3 i 4.5. za automobil V4, na kojima su prikazane srednje vrijednosti procjene brzine pomoću predloženog algoritma i njihove standardne devijacije u usporedbi sa izmjerenom srednjom vrijednosti brzine pomoću štoperice.

Tablica 4.4. Srednja vrijednost brzine dobivene algoritmom i standardna devijacija za automobil V3.

AUTOMOBIL V3		Brzina kretanja [km/h]		
	Srednja brzina kretanja	13.26	18.65	22.43
	Procijenjena srednja brzina ± standardna devijacija	14.08 ± 26.24	21.93 ± 25.80	26.24 ± 30.3

Tablica 4.5. Srednja vrijednost brzine dobivene algoritmom i standardna devijacija za automobil V4.

AUTOMOBIL V4		Brzina kretanja [km/h]		
	Srednja brzina kretanja	12.88	17.2	20.32
	Procijenjena srednja brzina ± standardna devijacija	18.40 ± 15.87	21.12 ± 27.12	25.71 ± 71.89

Iz priloženih rezultata vidljiv je jako visok stupanj raspršenosti rezultata. Razlog tome su velike oscilacije u visini BB-a pri testiranju. Standardna devijacija uglavnom je veća pri većim brzinama – razlog tome je što algoritam lošije računa brže promjene scene (kada se automobil kreće, za razliku od slučaja kada stoji), obzirom na to da uvijek radi na 15 FPS-a. Međutim, procjena srednje brzine dobro je izračunata za npr. automobil V3. To upućuje na to da se prilikom korištenja ovog algoritma za procjenu udaljenosti (odnosno brzine vozila) treba uzimati srednja vrijednost procjene kroz određeni vremenski period (npr. period od jedne sekunde) – ovisno koliko dozvoljava konačna aplikacija sa svojim zahtjevima (brzina kojom se očekuje da će se automobil kretati i učestalost obavještavanja vozača).

5. ZAKLJUČAK

Korištenje kamera kao zamjena za zrcalo u ADAS algoritmima kod novijih vozila sve je češća pojava. Omogućuju širi kut pregleda, veći broj mogućih pogleda, obavještavanje vozača o blizini drugog detektiranog vozila ili o potencijalnoj opasnosti ako se detektirano vozilo približava. U okviru rada razvijeno je rješenje koje koristi ukupno pet kamera postavljenih na maketu vozila. Razvijeno je rješenje sa ukupno četiri korisnička slučaja, a omogućavaju pregled okoline vozila i procjenu udaljenosti i brzine detektiranih vozila iza vozila. Za implementaciju korišteni su alati: programski jezik C, VISION SDK programsko okruženje, ADAS Alpha ploča, automotiv kamere i maketa vozila.

Predloženo rješenje sastoji se od četiri korisnička slučaja, pri čemu su omogućeni različiti pogledi okoline vozila. Kreiran je vlastiti algoritamski *link Mirror* koji u realnom vremenu izvršava zrcaljenje slike obzirom na središnju vertikalnu os slike na način da zamjenjuje odgovarajuće elemente slike iz originalnog signala s kamere. Prvi korisnički slučaj pokazuje okolinu pomoću tri širokokutne kamere, pri čemu je zbog kuta kamere obuhvaćena cijela okolina iza, a čak i bočno od vozača. U drugom korisničkom slučaju upotrebljavaju se jedna uskokutna i širokokutna kamera da bi se omogućila pomoć pri parkiranju vozila. Treći korisnički slučaj omogućuje vozaču odabir dijela prikazane slike kojeg želi uvećati, a pri tome koristeći dva stupnja uvećanja može izabrati manji ili veći dio slike kojeg želi uvećati. Brzina izvršavanja tih korisničkih slučajeva je oko 30 FPS-a, što odgovara FPS-u testnih kamera. Posljednji korisnički slučaj koristi algoritamske *linkove* za detekciju vozila koji su već implementirani u programskom okruženju VISION SDK. Na temelju detekcija automobila korišteni su parametri graničnih okvira (BB-a) kako bi se geometrijskim metodama izračunala udaljenost detektiranog automobila od kamere. Nakon utvrđivanja udaljenosti procjenjuje se i brzina kao promjena udaljenosti u jedinici vremena. Uz izračunatu vrijednost obrade jednog okvira, dodan je modul računanja brzine detektiranog vozila. Modul za računanje udaljenosti testiran je na udaljenostima od 5 do 30 metara, a modul za brzinu vozila pri brzinama od 15 do 25 km/h. Rezultati za procjenu udaljenosti postignuti su uz pravilna odstupanja - uz dodatni odgovarajući filter ili promjenu unaprijed zadane visine automobila rezultati se mogu dodatno korigirati. Rezultati za procjenu brzine imaju veliko raspršenje rezultata zbog oscilacija veličine BB-a, iako su srednje vrijednosti približne očekivanima. Uzrok tome je niska vrijednost FPS-a koja nije pogodna za pomične scene kao što je kretanje automobila.

LITERATURA

[1] „SAE International“. https://www.sae.org/standards/ (pristupljeno ruj. 07, 2020).

[2] S.-C. Lin *i ostali*, „The Architectural Implications of Autonomous Driving: Constraints and Acceleration“, u *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, Williamsburg VA USA, ožu. 2018, str. 751–766, doi: 10.1145/3173162.3173191.

[3] S. Dabral, S. Kamath, V. Appia, M. Mody, B. Zhang, i U. Batur, „Trends in camera based Automotive Driver Assistance Systems (ADAS)“, u *2014 IEEE 57th International Midwest Symposium on Circuits and Systems (MWSCAS)*, kol. 2014, str. 1110–1115, doi: 10.1109/MWSCAS.2014.6908613.

[4] B. Zhang *i ostali*, „A Surround View Camera Solution for Embedded Systems“, u *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, lip. 2014, str. 676–681, doi: 10.1109/CVPRW.2014.103.

[5] Y. Liu i B. Zhang, „Photometric alignment for surround view camera system“, u *2014 IEEE International Conference on Image Processing (ICIP)*, lis. 2014, str. 1827–1831, doi: 10.1109/ICIP.2014.7025366.

[6] M. Mody *i ostali*, „High Quality Image Processing System for ADAS“, u *2019 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, srp. 2019, str. 1–4, doi: 10.1109/CONECCT47791.2019.9012938.

[7] Y.-L. Chen i C.-A. Wang, „Vehicle Safety Distance Warning System: A Novel Algorithm for Vehicle Safety Distance Calculating Between Moving Cars“, u *2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring*, tra. 2007, str. 2570–2574, doi: 10.1109/VETECS.2007.529.

[8] M. Gluhaković, M. Herceg, M. Popovic, i J. Kovačević, „Vehicle Detection in the Autonomous Vehicle Environment for Potential Collision Warning“, u *2020 Zooming Innovation in Consumer Technologies Conference (ZINC)*, svi. 2020, str. 178–183, doi: 10.1109/ZINC50678.2020.9161791.

[9] *Joyride in the BMW i8 Mirrorless! - CES 2016*. 2016.

[10] *Audi Virtual Mirror Vs Lexus Digital Side-View Monitor*. 2018.

[11] *2020 Lexus ES DIGITAL SIDE MIRRORS and REVIEW – Lexus DIGITAL OUTER MIRRORS*. 2018.

[12] „2.5 Average car height, width and weight, 1980-2018“, *Mobility in Figures Cars 2019*. https://bovagrai.info/auto/2019/en/2-registrations/2-5-average-car-height-width-and-weight/ (pristupljeno ruj. 08, 2020).

SAŽETAK

Tema ovog rada je izrada ADAS algoritma u kojem se umjesto konvencionalnih zrcala u vozilima upotrebljavaju kamere. Najprije su predstavljena postojeća rješenja na temelju znanstvenih radova i primjeri iz automobilske industrije. Prikazane su različite funkcionalnosti povezane s kamerama kao zamjena za zrcalo kod nekoliko proizvođača automobila.

Predloženo rješenje se temelji na obradi signala koji se dobivaju sa pet različitih kamera, pri čemu se koriste tri širokokutne i dvije uskokutne kamere. Implementacija rješenja izvršena je u programskom okruženju VISION SDK koristeći ADAS Alpha ploču. Rješenje je napisano u programskom jeziku C. Implementirana su četiri korisnička slučaja i jedan algoritamski *link*. U korisničkim slučajevima s dvije i tri kamere prikazani su različiti prikazi okoline vozila. Takvi prikazi omogućuju pregled oba mrtva kuta oko vozila i vrlo dobar pregled ostatka okoline. Korisnički slučaj sa uvećavanjem željenog dijela slike omogućuje korisniku odabir dijela ekrana kojeg želi uvećati. Korisnik može odabrati između dvije razine uvećanja slike i prema tome se kreira prikaz na ekranu. Posljednji korisnički slučaj korišten je za detekciju automobila, izračun udaljenosti automobila od kamere i izračun brzine kretanja automobila. Ovaj korisnički slučaj temelji se na informaciji o visini graničnog kvadrata detekcije koji se dobiva od detektora automobila i na temelju kojeg se provodi proračun za udaljenost automobila. Procjena brzine izvršena je na temelju promjene udaljenosti kroz odgovarajući interval vremena. Verifikacija rada algoritama provedena je mjerenjem FPS-a, promatranjem opterećenja procesora pri izvođenju i uspoređivanjem dobivenih rezultata sa realnim. Nakon testiranja zaključeno je kako prva tri korisnička slučaja rade vrlo pouzdano. Korisnički slučaj sa detekcijom vozila i računanjem udaljenosti i brzine radi zadovoljavajuće. Rezultati potpuno ovise o algoritmu za detekciju automobila koji u realnim uvjetima ne radi uvijek pouzdano.

Ključne riječi : ADAS, TDA2xx, VISION SDK, kamere, zrcala, obrada slike

ABSTRACT

The topic of this paper is making an ADAS algorithm in which cameras are used instead of conventional mirrors in vehicles. First, the existing solutions are presented based on scientific papers and examples from the automotive industry. Various cameras-related functionalities are shown as a mirror replacement for several car manufacturers.

The proposed solution is based on the processing of signals received from five different cameras, using three wide and two narrow-angle cameras. The solution was implemented in the VISION SDK software framework using the ADAS Alpha board. The solution is written in the C programming language. Four use cases and one algorithm link were implemented. In use cases with two and three cameras, different views of the vehicle environment are shown. Such displays allow an overview of both blind spots around the vehicle and a very good overview of the rest of the surroundings. The use case with zooming in the desired part of the image allows the user to select the part of the screen he wants to enlarge. The user can choose between two different levels of zoom and can therefore be displayed on the screen. The last use case was used to detect the car, calculate the distance of the car from the camera and calculate the speed of the car. This use case is based on information about the height of the bounding box obtained from the car detector and on the basis of which the distance calculation of the car is performed. The speed estimate was performed based on the change in distance over an appropriate time interval. Verification of the operation of the algorithm was performed by measuring the FPS, observing the CPU load during execution and comparing the obtained results with the actual ones. After testing, it was concluded that the first three use cases work very reliably. The use case with car detection and calculating distance and speed works satisfactorily. The results depend entirely on the algorithm for detecting cars that is not always reliable in real conditions.

Keywords: ADAS, TDA2xx, VISION SDK, cameras, mirrors, image processing

ŽIVOTOPIS

Ivan Zovak rođen je 13. ožujka 1997. godine u Vinkovcima. Završio je Osnovnu školu „Ivan Kozarac“ u Županji. Nakon toga upisuje Prirodoslovno-matematičku gimnaziju u Županji koju uspješno završava 2015. godine. Iste godine upisuje preddiplomski smjer Elektrotehnika na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Na drugoj godini studija upisuje smjer Komunikacije i informatika, a na četvrtoj smjer diplomskog studija Automobilsko računarstvo i komunikacije. Trenutno pohađa drugu godinu diplomskog studija Automobilsko računarstvo i komunikacije i stipendist je Instituta RT-RK od 2019. godine.

_____

Potpis autora