Algoritam za dobivanje pogleda odozgo na vozilo

Marinić, Ivan

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

ALGORITAM ZA DOBIVANJE POGLEDA ODOZGO NA VOZILO

Diplomski rad

Ivan Marinić

Osijek, 2023. godina.

SADRŽAJ

1. UVOD

Računalni vid (engl. *Computer vision*) je područje koje se posljednjih godina značajno razvija, a tehnologije autonomne vožnje privlače veliku pozornost zbog svoje sposobnosti da revolucioniraju automobilsku industriju. Autonomna vozila moraju dobro razumjeti što se događa oko njih i donositi odluke na temelju razumijevanja okoline. Stoga, razvijanje algoritama i tehnika računalnog vida ima ključnu ulogu u osiguravanju sigurnog i učinkovitog upravljanja autonomnih vozila.

Napredni sustavi za pomoć vozaču (engl. *Advanced Driver Assistance Systems - ADAS*) omogućuju implementaciju različitih algoritama koji, putem prikupljenih podataka iz okoline vozila, olakšavaju vozaču kontrolu vozila i povećavaju sigurnost u prometu. Ti sustavi postižu sljedeće ciljeve: postupno zamjenjuju kontrolu upravljanja vozilom u određenim scenarijima bez potrebe za intervencijom vozača, upozoravaju na potencijalne opasnosti i prepreke te pružaju pregled okoline vozila. Visoka razina sigurnosti u prometu vozila postiže se zahvaljujući sve preciznijim i pristupačnijim senzorima, ograničenoj, ali dostatnoj računalnoj snazi ugrađenih procesora te razvoju algoritama koji rade u stvarnom vremenu.

Algoritam za dobivanje pogleda odozgo na vozilo omogućuje stvaranje konačne slike koja prikazuje prostor oko vozila iz ptičje perspektive. Ovaj algoritam se sastoji od nekoliko koraka koji uključuju radnje koje se izvode na ulaznim slikama dobivenim s različitih kamera na vozilu. Prvi korak algoritma je kalibracija kamera. Za to se koriste kalibracijski uzorci koji se postavljaju pred kamere. Na temelju tih uzoraka, algoritam izračunava parametre kamere koji su potrebni za uklanjanje izobličenja slika koje su snimljene širokokutnim kamerama. Nakon kalibracije kamera, slijedi proces obrade slike s ciljem dobivanja konačnog pogleda odozgo na vozilo. U ovom koraku se primjenjuju različite geometrijske operacije nad ulaznim slikama dobivenim s različitih kamera. Prvo se vrši korekcija izobličenja slika pomoću parametara dobivenih kalibracijom kako bi se ispravila bilo kakva izobličenja uzrokovana širokokutnim lećama kamera. Zatim se primjenjuje transformacija perspektive kako bi se slike snimljene s različitih kamera transformirale u pogled kao da su snimljene iz ptičje perspektive. Nakon što su slike ispravljene i perspektiva im je promijenjena, provodi se operacija spajanja slika. Ovdje se koriste tehnike spajanja slika kako bi se stvorila konačna slika koja prikazuje širi pogled

odozgo na vozilo i prostor oko njega. Spajanje slika obuhvaća precizno poravnanje slika i njihovo kombiniranje kako bi se stvorila jedinstvena slika pogleda odozgo.

U sklopu ovog diplomskog rada zadatak je bio pomoću dostupne makete autića i četiriju širokokutnih kamera, uz korištenje namjenske ugradbene platforme, snimiti okolinu vozila te izraditi algoritam za dobivanje pogleda odozgo na vozilo. Algoritam je razvijen na razvojnom računalu te nakon što je rješenje bilo funkcionalno, implementirano je na AMV Alpha ADAS razvojnoj ploči, gdje je demonstrirao sposobnost generiranja kvalitetne slike pogleda odozgo u stvarnom vremenu. Ovaj algoritam ima potencijalnu primjenu u raznim naprednim sustavima za pomoć vozaču i pomaže vozačima u boljem pregledu okoline vozila, čime se povećava sigurnost i olakšava upravljanje vozilom kao pomoć vozaču prilikom parkiranja vozila.

Ovim diplomskim radom proučene su i uspoređene postojeće metode za kalibraciju kamere i transformaciju perspektive slike, a tako i postojeća rješenje generiranja pogleda odozgo na vozilo. Također, istražene su i metode za ujednačavanje osvjetljenja na slikama. Na razvojnom računalu razvijen je algoritam te implementiran na razvojnoj ploči, nakon čega je testiran rad u stvarnom vremenu i ispitana je kvaliteta izvođenja algoritma.

Diplomski rad u drugom poglavlju iznosi teorijsku osnovu razvoja algoritma za dobivanje pogleda odozgo. Opisuje ključne pretpostavke te opis izvedbe algoritma i osvrt na postojeće algoritme za dobivanje pogleda odozgo. Definirani su i objašnjeni postupci geometrijskog i fotometrijskog poravnanja slika. U trećem poglavlju su opisane korištene tehnologije te je prikazan i objašnjen razvoj vlastitog programskog rješenja za dobivanje pogleda odozgo na vozilo. U četvrtom poglavlju dani su rezultati testiranja rada predloženog vlastitog rješenja. Na kraju rada iznesen je zaključak zasnovan na rezultatima testiranja.

2. PROBLEM DOBIVANJA POGLEDA ODOZGO NA VOZILO

Za razumijevanje algoritma za dobivanje pogleda odozgo na vozilo potrebno je poznavati teorijsku osnovu koja stoji iza generiranja pogleda. Kako bi ostvarivanje pogleda odozgo bilo moguće, algoritam mora proći kroz određene operacije obrade slike. Ključne komponente algoritma su: geometrijsko poravnanje, fotometrijsko poravnanje i generiranje cjelovitoga pogleda odozgo. Geometrijsko poravnanje ispravlja izobličenje slika nastalih snimanjem širokokutnim kamerama i prebacuje perspektivu slike u ptičju perspektivu. Slijedi algoritam za generiranje cjelovitog pogleda odozgo spajanjem susjednih prikaza. Konačno slijedi fotometrijsko poravnanje koje ispravlja neusklađenost svjetline i boje između susjednih prikaza kako bi se postiglo što bolje spajanje.

2.1. Geometrijsko poravnanje

Geometrijsko poravnanje odnosi se na procese ispravljanja izobličenja, transformacije perspektive te poravnanja slika snimljenih s različitih kamera koje pokrivaju različite strane vozila, kako bi se stvorio cjelovit i besprijekoran prikaz okoline vozila iz ptičje perspektive. Kamere su postavljene na točno određene pozicije na vozilu, pod određenim kutom, kako bi snimile područje ispred, iza te s bočnih strana vozila. Kao što je prikazano slikom 2.1. [21] u nastavku, kamere su postavljene na točno određene pozicije na automobilu te su nasuprotne kamere točno u ravnini jedna s drugom. Plavim kružićima su označeni položaji kamera na vozilu te je žutom bojom označeno područje preklapanja prikaza s kamera.

Slika 2.1. Područje prikaza kamera fiksiranih na vozilu [21]

Svaka kamera ima svoj vlastiti pogled. Ključnu ulogu u geometrijskom poravnanju ima rješavanje problema kalibracije kamere i uklanjanje izobličenja nastalih zbog optičkih svojstava širokokutnih kamera. Ova korekcija pomaže u očuvanju oblika i proporcija objekata okoline u pogledu odozgo. Precizno geometrijsko poravnanje osigurava da objekti i prostor oko vozila budu prikazani na točnoj i dosljednoj poziciji. To je ključno za pravilno funkcioniranje sustava za autonomnu vožnju, detekciju prepreka, praćenje okoline vozila i druge sigurnosne funkcionalnosti.

Tehnika geometrijskog poravnanja koristi matematičke transformacije, kao što su homografija i transformacija perspektive, kako bi se uskladile različite slike i ispravila perspektiva slika. Ovo omogućuje da se slike precizno preklapaju, stvarajući prikaz cjelovitog pogleda odozgo na vozilo. Ovaj postupak osigurava glatke prijelaze između slika i stvara koherentan i dosljedan pogled odozgo na cijelu okolinu vozila.

2.1.1. Izobličenje slike

Izobličenje slike je fenomen koji se javlja kada se projekcija stvarnog svijeta na sliku izobliči zbog optičkih svojstava kamere. Izobličena slika može utjecati na točnost i kvalitetu prikaza te može negativno utjecati na različite aplikacije iz područja računalnog vida, kalibracije kamere, navigacije i robotike. Dvije glavne vrste izobličenja su radijalno izobličenje i tangencijalno izobličenje.

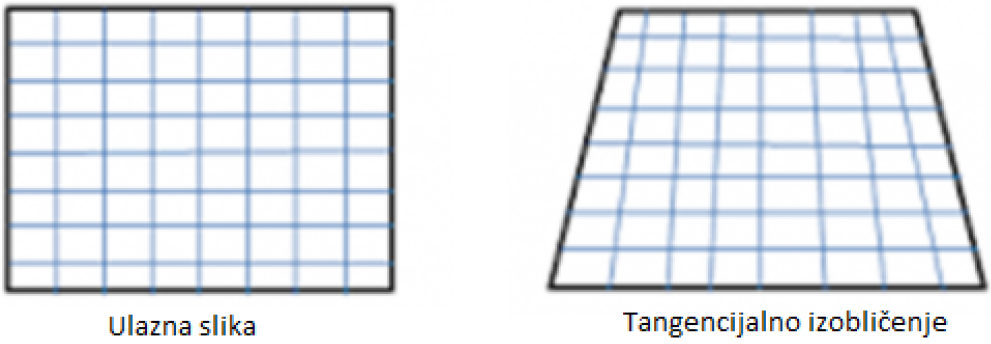
Radijalno izobličenje se obično javlja zbog nejednakog savijanja svjetlosti. Zrake se više savijaju u blizini rubova leće nego zrake u blizini središta leće. Zbog radijalnog izobličenja, ravne linije iz stvarnog svijeta na slici izgledaju zakrivljene. Svjetlosna zraka radijalno se pomiče prema unutra ili prema van od svoje idealne lokacije prije nego što udari u senzor slike. Postoje dvije vrste efekta radijalnog izobličenja, a to su:

- efekt bačvaste izobličenja (engl. *barrel distortion*), koji odgovara negativnom radijalnom pomaku
- efekt jastučastog izobličenja (engl. *pincushion distortion*), koji odgovara pozitivnom radijalnom pomaku.

Prikaz radijalnih izobličenja je na slici 2.2. [1] u nastavku.

Slika 2.2. Učinak bačvastog i jastučastog izobličenja na kvadratnoj mreži [1]

Tangencijalno izobličenje se obično događa kada je zaslon slike ili senzor pod određenim kutom u odnosu na leću. Stoga se čini da je slika nagnuta i rastegnuta. Prikaz tangencijalnog izobličenja je na slici 2.3. [15].



Slika 2.3. Učinak tangencijalnog izobličenja na kvadratnoj mreži [15]

Matematički, bačvasto i jastučasto izobličenje se povećava s kvadratom udaljenosti od središta. Radijalno izobličenje može se predstaviti na sljedeći način [2]:

$$x = x'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{2-1}$$

$$y = y'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{2-2}$$

$$r^2 = (x')^2 + (y')^2 \tag{2-3}$$

- x' i y' su neiskrivljene lokacije elemenata slike. x' i y' su u normaliziranim koordinatama slike. Normalizirane koordinate slike izračunavaju se iz koordinata elemenata slike prevođenjem u optički centar i dijeljenjem sa žarišnom duljinom u elementima.
- $k_1, k_2\ i\ k_3$ su koeficijenti radijalnog izobličenja leće.
- x i y su koordinate elementa slike u izobličenoj slici.
- r predstavlja udaljenost točke (x', y') od ishodišta koordinatnog sustava (0, 0).

Slično, dolazi do tangencijalnog izobličenja jer leća za snimanje slike nije savršeno paralelna s ravninom slike. Dakle, neka područja na slici mogu izgledati bliže od očekivanog. Količina tangencijalnog izobličenja može se predstaviti na sljedeći način [2]:

$$x = x' + [2p_1x'y' + p_2(r^2 + 2(x')^2)] \tag{2-4}$$

$$y = y' + [p_1(r^2 + 2(y')^2) + 2p_2x'y'] \tag{2-5}$$

- $p_1\ i\ p_2$ su koeficijenti tangencijalnog izobličenja leće.

Ukratko, potrebno je pronaći pet parametara, poznatih kao koeficijenti izobličenja koji su dati kao [2]:

$$koeficijenti\ izobličenja = (k_1, k_2, p_1, p_2, k_3) \tag{2-6}$$

Osim toga, potrebne su još neke informacije poput unutarnjih i vanjskih parametara kamere. Unutarnji parametri specifični su za kameru. Oni uključuju informacije poput žarišne duljine (f_x, f_y) i optičkog središta (c_x, c_y). Žarišna duljina i optičko središte mogu se koristiti za stvaranje matrice kamere, koja se može koristiti za uklanjanje izobličenja uzrokovanih lećama određene kamere. Matrica kamere **K** jedinstvena je za određenu kameru, pa se jednom izračunata može ponovno upotrijebiti na svim slikama snimljenim istom kamerom. Izražava se kao matrica 3x3 [2]:

$$\boldsymbol{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2-7}$$

Vanjski parametri kamere definiraju položaj i orijentaciju kamere u odnosu na referentni koordinatni sustav. Parametri uključuju rotacijske i translacijske parametre koji određuju kako je kamera postavljena u prostoru. Obično se predstavljaju matricom transformacije **T**, koja uključuje rotacijske i translacijske komponente i prikazana je jednadžbom (2-8) [16].

$$\boldsymbol{T} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \tag{2-8}$$

- $r_{11},\ r_{12},\ r_{13},\ r_{21},\ r_{22},\ r_{23}, r_{31},\ r_{32},\ r_{33}$ su elementi 3x3 rotacijske matrice koji predstavljaju rotacijske komponente transformacije kamere u trodimenzionalnom prostoru. Ovi elementi određuju kako se kamera rotira u prostoru i kako su njezine osi usklađene s referentnim koordinatnim sustavom. r_{11} predstavlja kosinus kuta između prve osi koordinatnog sustava kamere i prve osi referentnog koordinatnog sustava, r_{12} predstavlja kosinus kuta između prve osi koordinatnog sustava kamere i druge osi referentnog koordinatnog sustava te r_{13} predstavlja kosinus kuta između prve osi koordinatnog sustava kamere i treće osi referentnog koordinatnog sustava. Navedeni elementi predstavljaju rotaciju oko x osi kamere. Slijedno tome, r_{21} označava kosinus kuta između druge osi koordinatnog sustava kamere i prve osi referentnog koordinatnog sustava te predstavlja rotaciju oko y osi kamere. r_{31} označava kosinus kuta između treće osi koordinatnog sustava kamere i prve osi referentnog koordinatnog sustava te predstavlja rotaciju oko z osi kamere. Ostali elementi matrice analogno označavaju rotacije oko drugih osi referentnog koordinatnog sustava.

- $t_1,\ t_2,\ t_3$ su translacijski parametri koji predstavljaju pomak kamere u odnosu na referentni koordinatni sustav. Konkretno, t_1 označava pomak kamere duž prve osi referentnog koordinatnog sustava. Ovaj pomak može biti pozitivan ili negativan, ovisno o smjeru translacije. t_2 označava pomak kamere duž druge osi referentnog koordinatnog sustava. t_3 označava pomak kamere duž treće osi referentnog koordinatnog sustava.

2.1.2. Kalibracija kamere

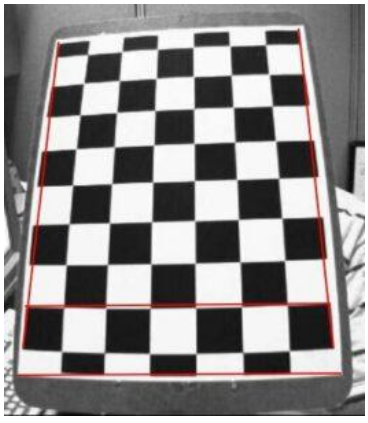
Kalibracija kamere je postupak određivanja unutarnjih parametara kamere i parametara izobličenja kako bi se omogućila precizna rekonstrukcija 3D prostora iz 2D slika. Ovi parametri su važni za ispravno mjerenje udaljenosti, kutova i dimenzija objekata na slici.

Nakon pojednostavljenja, matrica ukupne transformacije koja se koristi za ispravljanje izobličenja *fish-eye* kamere izgleda [19]:

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{pmatrix} \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \tag{2-9}$$

- s je parametar skaliranja koji određuje koliko će se izobličena slika razvući ili stisnuti tijekom ispravljanja.
- x_0 i y_0 su parametri pomaka (engl. *offsets*) koji određuju translaciju izobličene slike u x i y smjeru, što omogućuje centriranje ispravljene slike.

Leće svih kamera nisu jednake. Kako bi se moglo ispraviti izobličenje koje unosi leća, potrebno je poznavati parametre kamere. Najčešće se kalibracija radi za širokokutne kamere s *fish-eye* lećom. *Fish-eye* leća može biti pravocrtna, ortografska, stereografska ili ekvidistanta. Metoda za kalibraciju kamere iz [2] prikazana je pomoću uzorka šahovske ploče na slici 2.4. [2] u nastavku.



Slika 2.4. Uzorak šahovske ploče korišten za kalibraciju kamere u [2]

Može se reći da će svaka točka na šahovnici generirati matricu prikazanu jednadžbom (2-9). Jednadžba se može koristiti za definiranje matrice homografije **H** na sljedeći način [19]:

$$\boldsymbol{H} = \begin{pmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{pmatrix} \tag{2-10}$$

Najčešći pristup procjeni parametara izobličenja leće je njihovo modeliranje pomoću sljedećih jednadžbi [19]:

$$x' = x(1 + q_1 r^2 + q_2 r^4), \tag{2-11}$$

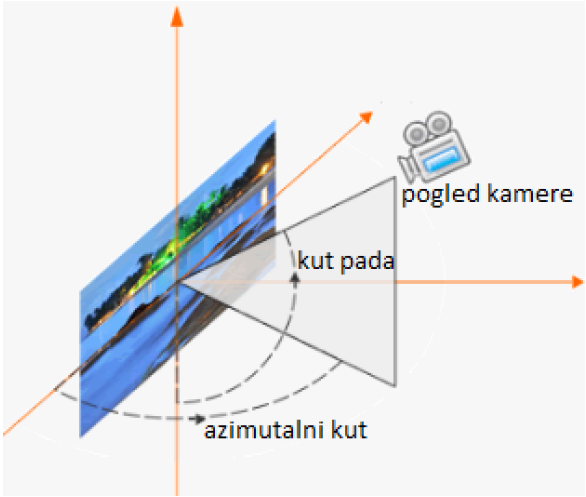
$$y' = y(1 + q_1 r^2 + q_2 r^4) \tag{2-12}$$

gdje r predstavlja udaljenost između elemenata slike i središta slike, dok q_1 i q_2 predstavljaju koeficijente izobličenja koji opisuju nelinearno izobličenje leće. Prethodne formule koriste se za prilagodbu točaka na izobličenoj slici kako bi se dobile njihove odgovarajuće točke na ispravljenoj slici.

Postupak kalibracije kamere uključuje postavljanje odnosa između 3D svijeta i 2D slike, kao i prilagodbu nepravilnosti leće kako bi se smanjilo izobličenje slike. Kalibracija kamere obično uključuje određivanje nelinearnih koeficijenata kako bi se precizno ispravila izobličenja i stvorila kvalitetna ispravljena slika. Važno je napomenuti da postoji više metoda kalibracije kamere, uključujući metodu direktnog proračuna, metodu samo-kalibracije, kao i metode zasnovane na umjetnoj inteligenciji i strojnom učenju.

2.1.3. Transformacija perspektive slike u pogled odozgo

Nakon kalibracije, koriste se dobiveni parametri za ispravljanje izobličenja. U nastavku su prikazane jednadžbe za kut pada (engl. *pitch*) α i azimutalni kut (engl. *yaw*) θ pomoću radijusa R, odnosno udaljenosti točke (x, y) od središta kamere, koji omogućuju precizno određivanje orijentacije objekta u prostoru u odnosu na kameru, što je ključno za transformaciju perspektive. Azimutalni kut predstavlja kut rotacije oko vertikalne uzdužne osi kamere. Ovaj kut određuje horizontalni smjer gledanja kamere u odnosu na referentni smjer. Kut pada predstavlja kut rotacije oko horizontalne bočne osi kamere. Ovaj kut određuje vertikalni smjer gledanja kamere prema dolje prema sceni. U nastavku su navedeni kutovi skicirani na slici 2.5. [20], dok slika 2.6. [20] predstavlja promjenu kutova na snimljenoj slici.



Slika 2.5. Prikaz pogleda kamere s kutom pada i azimutalnim kutom [20]

a) b)

Slika 2.6. Postupak promjene kutova na snimljenoj slici (a) originalna snimljena slika (b) slika nakon promjene kuta pada za 30° i azimutalnog kuta za 25° [20]

U nastavku su formulama (2-13) i (2-14) prikazani izračuni kutova [17]:

$$\alpha = arcta\,n\left(\frac{\sqrt{x^2+y^2}}{R}\right), \tag{2-13}$$

$$\theta = arctan\,(y/x) \tag{2-14}$$

Neka je r označen kao udaljenost između središta ispravljene slike i drugih elemenata slike u ispravljenoj slici bez izobličenja. Formule od (2-15) do (2-17) u nastavku prikazuju izračun udaljenosti i izračun novih elemenata ispravljene slike [17].

$$r = R \cdot (\alpha/90°), \tag{2-15}$$

$$x' = r \cdot cos\,(\theta), \tag{2-16}$$

$$y' = r \cdot sin\,(\theta) \tag{2-17}$$

Tako se dobije odnos mapiranja između (x, y) i (x', y').

Transformacija perspektive je projiciranje slike na novu ravninu gledanja, također poznato kao projektivno mapiranje (engl. *projective mapping*). Da bismo dobili sliku odozgo iz slike bez izobličenja, koristi se metoda koja odabire četiri karakteristične točke na slikama bez izobličenja i njihove odgovarajuće pozicije na slikama odozgo. Izračunava se matrica homografije između tih dviju ravnina, a zatim se postiže transformacija perspektive. Opći model transformacije perspektive je [17]:

$$\rho[x' \quad y' \quad w']^T = \boldsymbol{H}[x \quad y \quad w]^T \tag{2-18}$$

Neka $(\mathrm{x}, \mathrm{y}, \mathrm{w})$ budu koordinate karakterističnih točaka u izvornoj slici s *w* homogenim koeficijentom koji predstavlja proširenje za prebacivanje u 2D prikaz, dok su (x', y', w') koordinate odgovarajućih točaka u slici nakon transformacije perspektive. Matrica **H** je matrica homografije između izvorne i odredišne ravnine, a ρ je faktor skaliranja. Matrica homografije te računanje novih koordinata pomoću iste prikazano je u nastavku [17]:

$$\boldsymbol{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}, \tag{2-19}$$

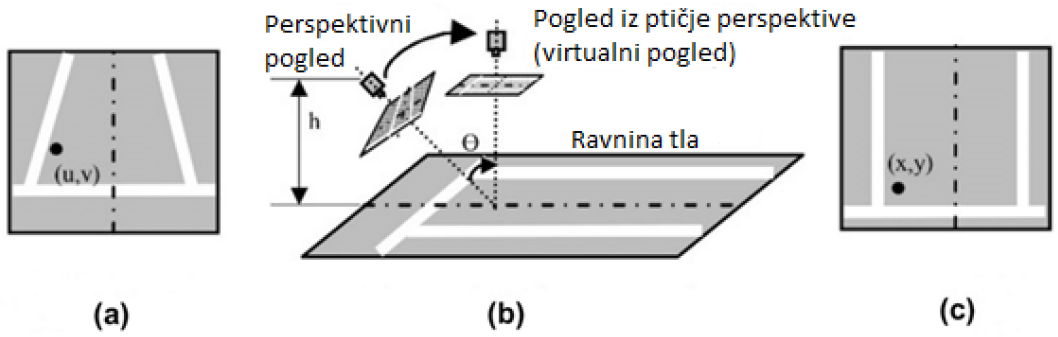
$$\rho[x' \quad y' \quad w']^T = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} [x \quad y \quad w]^T, \tag{2-20}$$

$$u = \frac{x'}{w'} = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \tag{2-21}$$

$$v = \frac{y'}{w'} = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \tag{2-22}$$

Oznake u i v predstavljaju koordinate transformirane točke u homogenom koordinatnom sustavu nakon primjene homografije. Matrica homografije se normalizira tako da w' iznosi 1.

Dakle, $u = x', v = y'$. Prolaskom kroz svaku (x, y) koordinatu slike, računaju se nove koordinate za transformaciju perspektive slike. Koordinate izračunate po formulama (2-21) i (2-22) predstavljaju koordinate transformiranih elemenata slike. U nastavku slika 2.7. [4] predstavlja prikaz primjene matrice homografije te promjenu perspektive ulazne slike.



Slika 2.7. Postupak primjene transformacije perspektive (a) ulazni prikaz (b) snimani i željeni prikazi (c) prikaz nakon promjene perspektive slike u pogled odozgo [4]

2.2. Fotometrijsko poravnanje

Fotometrijsko poravnanje je tehnika koja se koristi u algoritmima za dobivanje pogleda odozgo na vozilo kako bi se osigurala konzistentnost osvjetljenja i boje između različitih slika koje su snimljene s različitim kamerama. Uklanjaju se varijacije svjetline i boje između susjednih slika. To se može postići primjenom različitih transformacija, poput histogramskog poravnanja ili korištenja srednje vrijednosti.

2.2.1. Korekcija osvjetljenja

Osnovna metoda linearnog transformiranja koristi kombinaciju operacija množenja i zbrajanja nad vrijednostima elemenata slike, kako bi se manipuliralo razinom svjetline i boje slike. Ova metoda također je poznata kao poboljšanje svjetline i kontrasta. Kombinira dvije operacije u jednu, pri čemu parametri za množenje i zbrajanje određuju stupanj promjene svjetline i kontrasta slike. Parametar za množenje često se naziva faktorom pojačanja (engl. *gain*), a za zbrajanje se naziva parametrom pomaka (engl. *bias*). Odnos između originalne i konačne svjetline elemenata slike izražava se kako je prikazano u jednadžbi (2-23) [18]:

$$g(i,j) = \alpha \times f(i,j) + \beta \tag{2-23}$$

gdje (i,j) predstavlja vrijednosti koordinata elemenata slike, $g(i,j)$ predstavlja vrijednost svjetline elementa izlazne slike, $f(i,j)$ predstavlja vrijednost svjetline elementa ulazne slike, α predstavlja parametar pojačanja, a β predstavlja parametar pomaka. Vrijednosti α i β određuju razinu promjene svjetline između ulazne i izlazne slike. Budući da ovi parametri određuju stupanj promjene svjetline, također bi trebali biti određeni na temelju svjetlosne razine ulaznih slika. Parametri pojačanja i pomaka određuju se izračunom srednje svjetlosne razine referentnih i ulaznih slika u preklapajućem području. Nakon što se izračuna srednja vrijednost ulaznih slika, uzima se omjer svjetline tamnije slike prema svjetlijoj slici, koji je uvijek manji od jedan i koristi se kao *gamma* (γ).

Parametri pojačanja i pomaka izračunavaju se kako je prikazano u jednadžbama (2-24) do (2-26) [18].

$$\gamma = \left(\overline{Y_{dark}} / \overline{Y_{bright}}\right), \tag{2-24}$$

$$\alpha = 2 - \gamma, \tag{2-25}$$

$$\beta = \gamma \left(\overline{Y_{bright}} - \overline{Y_{dark}}\right) / 2 \tag{2-26}$$

gdje $\overline{Y_{\text{dark}}}$ predstavlja srednju vrijednost svjetline tamnije slike u preklapajućem području, $\overline{Y_{\text{bright}}}$ predstavlja srednju vrijednost svjetline svjetlije slike u preklapajućem području, γ predstavlja omjer *gamma* za korekciju svjetline i određivanje parametara pojačanja i pomaka, α predstavlja parametar pojačanja, a β predstavlja parametar pomaka.

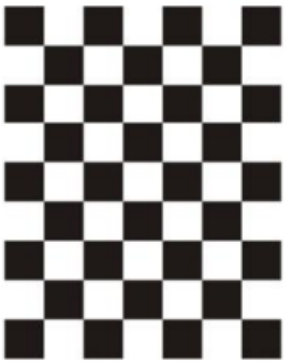
2.3. Pregled postojećih algoritama za dobivanje pogleda odozgo na vozilo

Osvrtom na radove obuhvaćaju se različite predložene ideje i rješenja dobivanja pogleda odozgo na vozilo. Izrada vlastitog algoritma zasniva se na već izrađenim rješenjima. Cilj i zadatak je izraditi rješenje koje će biti implementirano na stvarnoj ugradbenoj platformi kakva

se koristi u današnjim suvremenim automobilima. U nastavku su prikazana i objašnjena neka postojeća rješenje te su analizirane njihove prednosti i nedostaci.

U radu [6] fokus je na zahtjevima stvarne primjene ugrađenog pogleda odozgo na vozilu u realnom vremenu kako bi se pomoglo pri parkiranju. Uvodi se metoda kalibracije i alat za kalibraciju koji je dizajniran za opću namjenu na računalu, kao i tablice mapiranja podataka. Na kraju se stvara panorama koja može generirati sliku koristeći tablice mapiranja. Kako bi se riješili nedostaci postojeće tehnologije, uvodi se metoda kalibracije. Alat za kalibraciju je dizajniran na računalu i koristi se za kalibraciju ugrađenog sustava. Korištenjem dostupnih resursa, poput biblioteke *OpenCV* (engl. *Open Source Computer Vision Library*), programerima je omogućeno korištenje naprednih mogućnosti obrade slika. Rješenje se sastoji od dva podsustava: podsustava kalibracije i podsustava za generiranje panorame. Koriste se izvorne slike snimljene sa širokokutnim kamerama koje se obrađuju i mapiranjem podataka između izvorne slike i panoramske slike dobiva se pogled odozgo. Procesiraju se četiri slike, obrađuju i spajaju, a rezultat se prikazuje na zaslonu. Ovakvo rješenje omogućuje precizno generiranje pogleda odozgo u stvarnom vremenu za pomoć pri parkiranju vozila. Kalibracija sustava i korištenje naprednih alata i resursa kao što je *OpenCV* pridonose uspješnoj implementaciji i poboljšanju postojeće tehnologije.

Metoda je dizajnirana na računalu opće namjene u okruženju *VC++* i *OpenCV*. Za kalibraciju širokokutne kamere koristi se *OcamCalib Toolbox* za *MATLAB*. Metoda zadovoljava zahtjeve stvarnog vremena koji zahtijevaju dovoljno visok broj okvira po sekundi (engl. *Frames per second* - FPS) kako bi prikaz slike bio kontinuiran, oko 10 okvira po sekundi s minimalnom latencijom te rezolucijom slike koja se temelji na slikama koji se pružaju kao ulaz i koje moraju biti dovoljno visoke rezolucije kako bi pružile jasnu percepciju okoline. Uvedena je metoda kalibracije kako bi se poboljšala postojeća tehnologija, uz smanjenje troškova i razvojnih poteškoća. Kalibracija je omogućena pomoću uzorka kalibracije kamere koji prikazuje uzorak šahovnice te prikazan je slikom 2.8. [6].

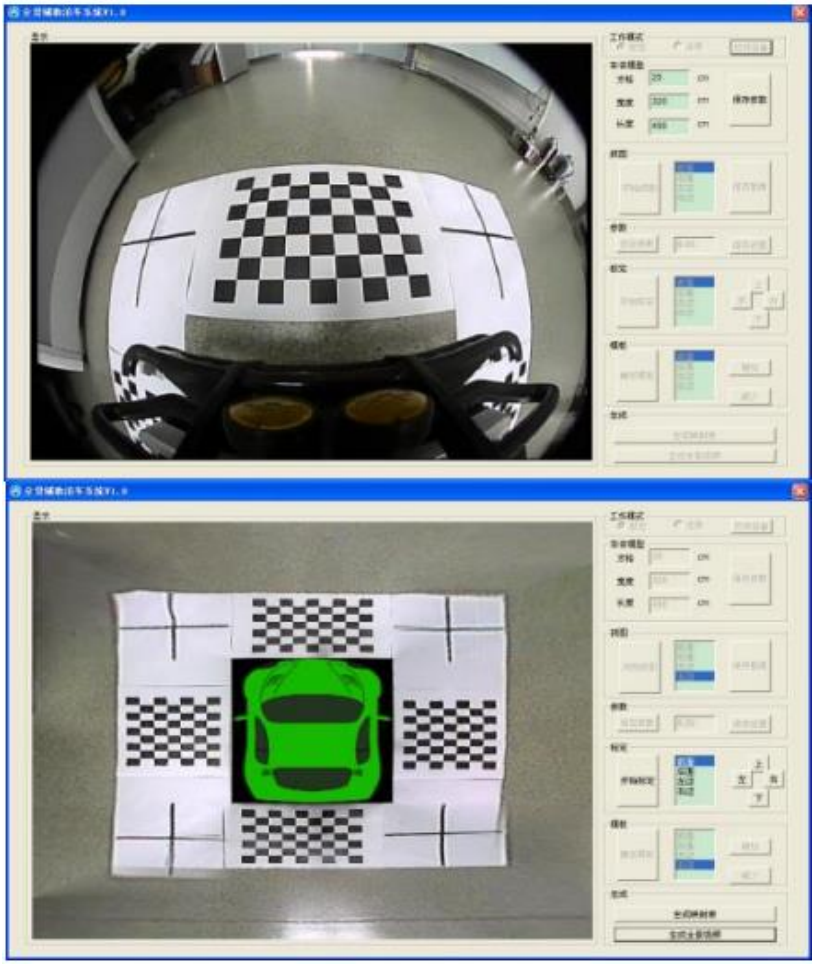


Slika 2.8. Uzorak za kalibraciju kamere korišten u [6]

Izrađen je i alat za kalibraciju koji izravno služi za ispravljanje i dobivanje krajnjeg pogleda odozgo na vozilo. Ulazne slike se ispravljaju, a položaj četiriju setova slika prilagođava se u panoramskoj slici kako bi se dobila potpuna panoramska slika oko vozila.

Izrađeni alat je prikazan na slici 2.9. [6]. Prvi prozor prikazuje korištenje *OcamCalib Toolbox*-a za kalibraciju prednje kamere automobila s *fish-eye* lećom. Učitava se slika s prednje kamere automobila i unose se parametri za ispravljanje izobličenja i prebacivanje perspektive kako bi se dobio pogled odozgo na vozilo. Taj proces se izvodi za slike sa svih četiriju kamera. Drugi prozor prikazuje spajanje četiriju ispravljenih i transformiranih slika koje su ranije učitane. Označava se koja slika je na kojoj strani automobila kako bi se pravilno povezali rubovi prikaza. Četiri slike se spajaju kako bi se dobio jedinstveni prikaz okoline automobila iz ptičje perspektive.

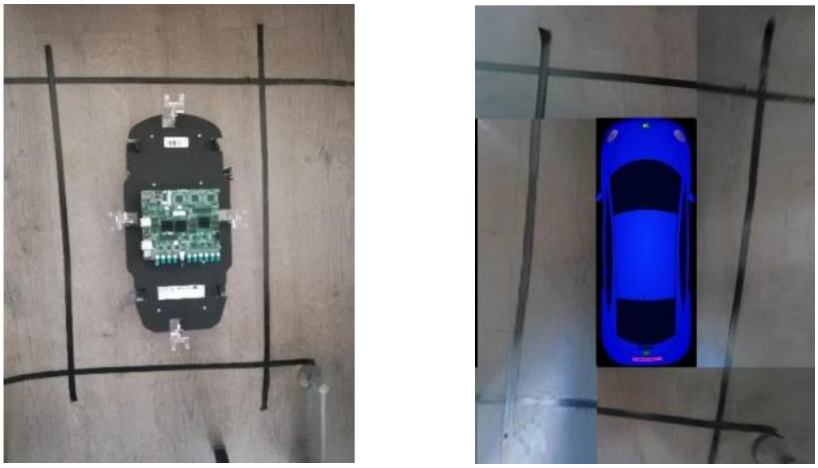
Budući da se koristi nekoliko širokokutnih kamera koje ne moraju biti visoke razlučivosti i kvalitete, može doći do većih deformacija slika i smanjene kvalitete. Također, udaljenost prikazana na zaslonu često odstupa od stvarne udaljenosti, što može otežati procjenu udaljenosti vozaču. Unatoč određenim nedostacima, ova metoda pruža korisnu podršku vozačima pri parkiranju vozila. Kalibracija kamera i upotreba naprednih alata poput *OpenCV* omogućuju poboljšanje kvalitete slika i precizniju procjenu udaljenosti. Važno je napomenuti da je ovo rješenje implementirano za sustav na računalnoj platformi, a ne za stvarni autonomni sustav u automobilu. Uz kontinuirani razvoj i daljnje poboljšanje tehnologije, ova metoda može postati još učinkovitija i pouzdanija u stvarnom svijetu.



Slika 2.9. Kalibracijski alat korišten u [6]

U radu [7] se za generiranje slike odozgo koristi AMV Alpha ADAS ploča i četiri prave automobilske širokokutne kamere. Prvo se kalibriraju kamere koristeći standardnu šahovsku ploču kako bi se ispravio *fish-eye* efekt širokokutne kamere. Nakon toga se primjenjuju određene operacije na snimljenim slikama kako bi se dobio konačan pogled odozgo. Ovi koraci uključuju stvaranje novih kalibracijskih slika i precizniju kalibraciju. Rješenje uključuje geometrijsko poravnanje putem ispravljanja izobličenja širokokutne kamere *LDC* (engl. *Lens Distortion Correction*) funkcijom radijalnog izobličenja i transformacije perspektive. Također se primjenjuje globalno dizajnirana funkcija za korekciju boje i svjetlosti. Izvršava se transformacija perspektive i usklađivanje slika kako bi se omogućilo njihovo spajanje za stvaranje pogleda iz ptičje perspektive. Za ovo istraživanje koristila se ADAS Alpha ploča s pratećim *VisionSDK* softverom i RDACM24B kamerama. Kalibracija kamere se provodi

pomoću šahovske ploče kao uzorka i snimanjem 15 kalibracijskih slika s različitim položajima i usmjerenjima kalibracijskog panela. Algoritam za kalibraciju kamere razvijen je u *Python* programskom jeziku koristeći *OpenCV* biblioteku. Nakon kalibracije kamere, generiranje izlazne slike odozgo omogućeno je pomoću algoritma na blokovskoj shemi razvijenog u *C++* koristeći *OpenCV* biblioteku. Predloženo rješenje omogućava udobnije, jednostavnije i sigurnije parkiranje vozila, posebno pri kretanju unatrag. Rezultati su pokazali vrlo vjerodostojne prikaze vozila s nekim manjim neusklađenostima. U nastavku, na slikama 2.10. i 2.11. [7] prikazane su referentne slike i dobivene slike rješenja za dvije različite okoline vozila.



a) b)

Slika 2.10. Prikaz okoline vozila za prvu scenu (a) referentna slika (b) prikaz slike rješenjem iz [7]

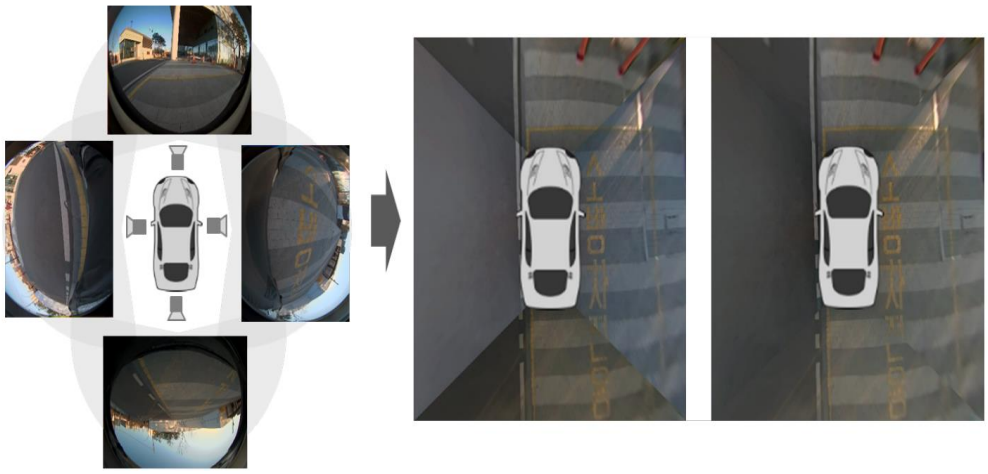
a) b)

Slika 2.11. Prikaz okoline vozila za drugu scenu (a) referentna slika (b) prikaz slike rješenjem iz [7]

Nedostaci predloženog rješenja su nesavršenost u spajanju susjednih slika nastale prilikom izrade i izvođenja rješenja i procesa dobivanja izlazne slike. Na rubovima u određenim testovima se vidi fotometrijsko i geometrijsko odstupanje. Rješenje za generiranje pogleda prostora oko automobila uspješno je razvijeno na računalu koje ispunjava sve tehničke zahtjeve potrebne za izvođenje algoritma. Međutim, rješenje nije uspješno implementirano na Alpha razvojnu ploču. Razlog tome su ograničenja memorije za pohranu i obradu velikih količina podataka te nedostatna programska podrška na razvojnoj ploči, što rezultira nedovoljnom implementacijom svih potrebnih modula za algoritme opisane u radu.

Sustav za nadzor okruženja predstavljen u [8] pruža kompozitni pogled na vozilo iz ptičje perspektive radi pomoći pri sigurnom parkiranju. Svaka od četiriju kamera automatski prilagođava ekspoziciju i balans bijele boje. Predloženo rješenje omogućuje uspješno spajanje četiriju slika uz učinkovito poravnanje za sustav nadzora pomoću jednostavnog modela aditivnih dobitaka. Aditivni dobitci su termin koji se koristi u kontekstu digitalne obrade slike i fotografije. Ova tehnika se koristi kako bi se poboljšala kvaliteta slika koje su snimljene u uvjetima s nedostatkom svjetlosti, što može rezultirati tamnijim i manje detaljnim slikama.

Rješenje se sastoji od tri koraka. Prvi korak uključuje odabir uzoraka podataka iz svake ulazne slike i izdvajanje statistika koje će se koristiti za odabir referentnog prikaza i procjenu dobitaka. Zatim se odabire referentni prikaz koji će biti osnova za određivanje konačne svjetline i boje. Konačno, procjenjuju se dobitci pomoću prikupljenih statistika. Predložena metoda je implementirana na NVIDIA Tegra CX ugrađenoj platformi, a vrijeme obrade je izuzetno brzo, tj. 3 ms koristeći grafičku karticu i procesor računala za rezoluciju ulaznih slika 1280 x 960 elemenata slike. Ostvareno rješenje ima mogućnost obrade i prikaza 30 FPS. Ovaj pristup pruža učinkovito rješenje za fotometrijsko poravnanje slika za sustav nadzora iz ptičje perspektive pomoću jednostavnog modela. Osigurava uspješno spajanje slika iz ptičje perspektive uz minimalne računalne zahtjeve. Važno je napomenuti da je ovo rješenje implementirano za sustav na računalnoj platformi, a ne za autonomni sustav u automobilu koji zahtjeva manju složenost algoritma zbog veće ograničenosti resursima. U nastavku je slikom 2.12. [8] prikazan proces ostvarivanja jedinstvenog pogleda odozgo na vozilo.



Slika 2.12. Prikaz postupka ostvarivanja krajnjeg pogleda odozgo rješenjem predstavljenim u [8]

2.4. Osvrt na postojeća rješenja

Analizom postojećih rješenja utvrđen je pristup koji će se koristiti pri izradi vlastitog algoritma za dobivanje pogleda odozgo na vozilo. U nastavku je iznesen kritički osvrt na radove predstavljene u prethodnom potpoglavlju.

Kada se razmotri rad [6] primjećuje se da je fokusiran na fotometrijsko poravnanje sustava kamera s višestrukim pogledima. Njegova prednost leži u upotrebi fotometrijske analize za poboljšanje kvalitete slika i preciznosti rekonstrukcije okoline. Korištenje naprednih alata poput *OpenCV*-a omogućuje kontinuirano prikazivanje slike s minimalnom latencijom. Kalibracija kamere doprinosi poboljšanju kvalitete slike snimljene širokokutnom kamerom i vjerniji prikaz okoline vozila. Međutim, istovremeno se suočava s nekim glavnim nedostacima, poput osjetljivosti na promjene osvjetljenja i mogućnosti poteškoća u obradi slika s izraženim šumom. Korištenje širokokutne kamere može dovesti do većih izobličenja slika te se prikazana udaljenost na prikazu može razlikovati od stvarne udaljenosti, što može otežati procjenu udaljenosti vozaču. Rješenje je izvedeno na razvojnom računalu te zahtjeva veću procesorsku moć i dostupnu memoriju, što predstavlja ograničenje za implementaciju sustava na stvarni sustav za pomoć vozaču s ograničenim resursima.

Zatim, u radu [7] predstavlja se metoda kalibracije kamera za sustav pomoći pri parkiranju s višestrukim pogledom. Njegova praktičnost i točnost kalibracije predstavljaju ključne

prednosti koje mogu poboljšati iskustvo vozača. Rezultati dobivenih prikaza pokazuju vrlo vjerodostojne prikaze okoline vozila, s manjim neusklađenostima. Korištenjem *OpenCV* biblioteke omogućena je obrada slika prikaza okoline vozila. Međutim, postoje i određeni nedostaci. Spajanje susjednih slika može biti nesavršeno i rezultirati fotometrijskim i geometrijskim odstupanjima na rubovima. Također, ograničenja memorije i nedovoljna programska podrška na Alpha razvojnoj ploči otežavaju potpunu implementaciju svih potrebnih modula algoritama opisanih u radu tako da je potpuno rješenje izvedeno samo na razvojnom računalu.

Treći rad, [8], predstavlja algoritam za sustav pomoći pri parkiranju s višestrukim pogledom. Njegova ključna prednost je izuzetno brza obrada slika na NVIDIA Tegra CX ugrađenoj platformi, što omogućuje obradu i prikaz 30 FPS za rezoluciju od 1280 x 960 elemenata slike ulaznih slika. Također, pristup koristi jednostavan model aditivnih dobitaka i osigurava minimalne računalne zahtjeve za spajanje slika. Međutim, važno je napomenuti da je ovo rješenje implementirano na računalnoj platformi i da nije implementirano u autonomnim sustavima vozila. U tom smislu, potrebno je prilagoditi algoritam i smanjiti složenost kako bi se omogućila primjena u sustavima za pomoć vozaču s ograničenim resursima.

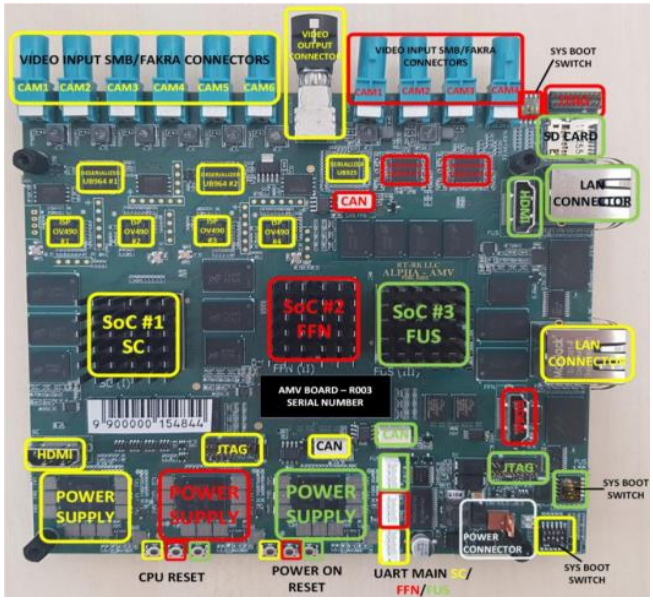
U cjelini, postojeća rješenja koja su analizirana u ovom poglavlju pružaju uvid u fotometrijsko poravnanje, kalibraciju i algoritme za obradu slika u sustavima s višestrukim pogledom za pomoć pri parkiranju. Njihove prednosti i nedostaci služe kao osnova za daljnji razvoj vlastitog rješenja. Kombiniranjem najboljih karakteristika ovih radova i uzimajući u obzir njihove izazove, teži se ka stvaranju naprednog i pouzdanog sustava za pomoć pri parkiranju koji će se moći primijeniti u različitim uvjetima, sustavima i vozilima.

3. VLASTITI ALGORITAM ZA DOBIVANJE POGLEDA ODOZGO NA VOZILO

Zadatak rada bio je razviti programsko rješenje na osobnom računalu te ga implementirati na Alpha ADAS razvojnu ploču pomoću *C* programskog jezika unutar *VisionSDK* programskog okruženja. U nastavku ovog poglavlja dano je nešto više detalja o razvojnoj ploči, programskom okruženju, tehnologijama korištenih tokom izrade i samim koracima razvoja programskog rješenja.

3.1. Opis razvojne ploče za napredne sustave pomoći vozaču na koju je potrebno implementirati algoritam

AMV Alpha *Board* [9] je ADAS razvojna ploča koju je dizajnirala tvrtka RT-RK. Ploču pokreću TI TDA2xx *SoC*-ovi (engl. *Socket on Chip*) i sadrži razne periferije. Uz ploču, *VisionSDK* se koristi kao razvojni softver. AMV Alpha je ploča koja sadrži 3 Texas Instruments *SoC*-a i podržava osnovne i napredne sustave upozorenja, aktivne sustave upravljanja i poluautonomne operacije za ADAS i automobilske aplikacije. Skup aplikacija pokriva različite upravljačke programe pomoći u vožnji kao što su pogled sprijeda, prikaz okoline vozila, nadzor vozača i noćni vid, s raznim značajkama za povećanu udobnost i opće iskustvo vožnje. Izgled razvojne ploče AMV Alpha ADAS prikazan je na slici 3.1. [9].

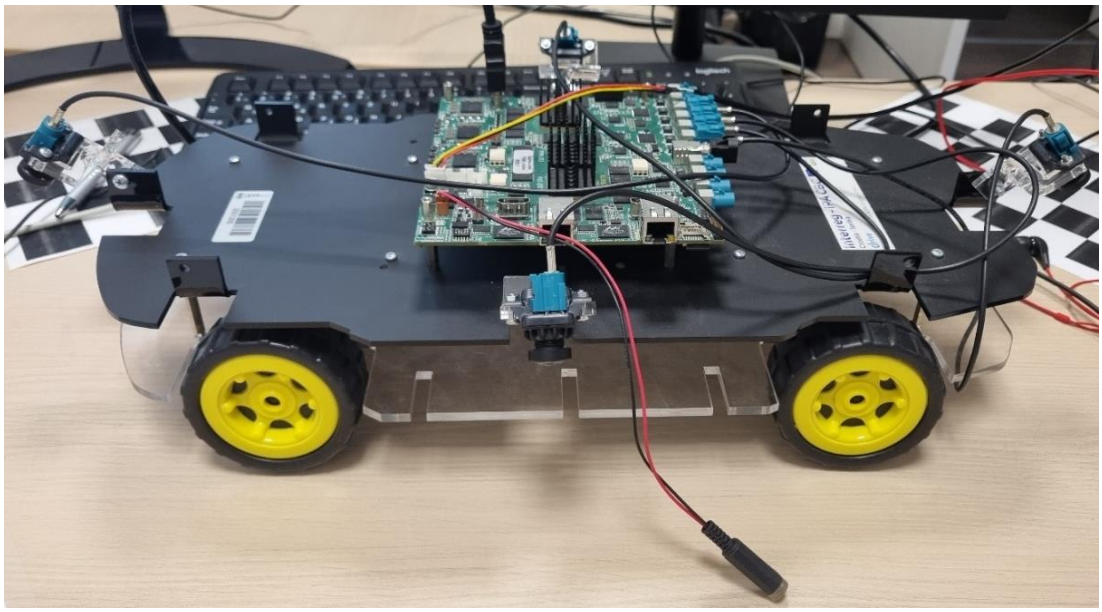


Slika 3.1. Izgled AMV Alpha ADAS razvojne ploče [9]

Postoje tri *SoC*-a, SC, FFN i FUS na AMV Alpha ploči. Prvi SC *SoC* primjenjuje se za aplikacije prikaza okoline vozila (engl. *Surround Camera View - SCV*). Drugi *SoC* FFN omogućuje prikaz kamera s prednjim pogledom, stereoskopski pogled bliskog kuta te noćni vid. Posljednji *SoC* koji se koristi je FUS. Potrebno je distribuirati resurse za obradu podataka između SCV i FFN *SoC*-a pomoću veze *Component Interconnect Express* (*PCIe*) i video veza između FFN i FUS *SoC*-ova. Svaki *SoC* ima sljedeće resurse i periferne uređaje na raspolaganju:

- 2x ARM Cortex-A15 CPU za korištenje mrežnih funkcija;
- 2x ARM Cortex-M4 CPU za opću namjenu koji se najviše koristi za slučajeve upotrebe i veze;
- 2x C66x DSP CPU za obradu slike visokih performansi;
- 2x EVE CPU za potrebe vezane uz vid;
- 1,5 GB RAM-a;
- Utor za mikro SD karticu za pokretanje kompilirane slike, čitanje podataka s kartice ili pisanje;
- HDMI priključak za prikaz video izlaza na ekranu;
- UART priključak za komunikaciju preko terminala na računalu;
- JTAG priključak za povezivanje sa sondom za otklanjanje pogrešaka;
- Prekidač načina pokretanja se koristi za podešavanje načina u kojem će se *SoC* pokrenuti;
- *PCIe* veza s drugim *SoC*-ovima za slanje podataka između *SoC*-ova.

Razvojna ploča je postavljena i fiksirana na maketu autića s priključenim i fiksiranim četirima širokokutnim kamerama. Maketa autića prikazana je slikom 3.2.



Slika 3.2. Izgled makete autića s razvojnom Alpha ADAS pločom korištene u izradi ovog rada

3.2. Programsko okruženje i biblioteke korištene za razvoj algoritma

Za razvoj vlastitog algoritma korištena je AMV Alpha ADAS ploča postavljena na maketu autića s fiksiranim četirima širokokutnim kamerama sa svih strana. Ploča se povezuje s osobnim računalom na kojemu se razvija programsko rješenje. Nakon što je rješenje razvijeno na računalu, implementira se na razvojnu ploču i testira se njegova funkcionalnost.

3.2.1. Alati i tehnologije korištene za izradu programskog rješenja

Za razvoj rješenja za primjenu na ADAS Alpha ploči korišteni su *Code Composer Studio,* integrirano razvojno okruženje tvrtke Texas Instruments. *Code Composer Studio* je *IDE* (engl. *Integrated Development Environment*) kao programsko okruženje koje podržava njihov portfolio mikrokontrolera i ugrađenih procesora. Sadrži alate za razvoj i ispravljanje pogrešaka ugrađenih aplikacija, uključujući optimizacijski *C/C++* kompajler, uređivač izvornog koda, okruženje za izgradnju projekata i ispravljanje pogrešaka.

Također je korišten *Python* programski jezik za pisanje izvršnih *Python* skripti u *Visual Studio Code*-u (*VS Code*) koji predstavlja integrirano razvojno okruženje s integriranim raznim naprednim funkcijama. *VS Code* nudi podršku za integraciju raznih *Python* alata korištenih za

izradu rješenja na razvojnom računalu. Radno okruženje s *Python* verzijom 3.11.1, integriranim razvojnim i edukacijskim okruženjem te upraviteljem *Python* paketa *PIP* bili su osigurani pomoću *Python installera*. Dodatno je nadograđen *Python* s paketima biblioteke *OpenCV*, *NumPy*, *PIL (Python Imaging Library)* i *imutils* putem *PIP*-a.

Tera Term je besplatan terminalni emulator i komunikacijski program koji se koristi za povezivanje računala s razvojnom pločom putem serijskog priključka. Omogućuje interaktivnu komunikaciju, podržava različite komunikacijske protokole i pruža korisne značajke za slanje i primanje podataka. *Tera Term* pruža interaktivno sučelje za slanje i primanje podataka te automatsko ponavljanje naredbi, podešavanje parametara veze (poput brzine prijenosa, bitova podataka, stop bitova itd.).

Za pokretanje YUV videozapisa snimanih pomoću RDACM24B-01 širokokutnih kamera povezanih s razvojnom pločom korišten je *YUVPlayer*. *YUVPlayer* je multimedijski reproduktor koji se koristi za prikazivanje i reprodukciju videozapisa u YUV formatu. Omogućuje korisnicima da otvore YUV videozapise i prikažu sliku u stvarnom vremenu. Također, pruža mogućnost pregledavanja pojedinačnih okvira unutar videozapisa, pomjeranja kroz video i kontroliranja brzine reprodukcije.

3.2.2. Programske biblioteke

OpenCV je biblioteka otvorenog koda za računalni vid i strojno učenje. Pruža zajedničku infrastrukturu za programe računalnog vida i ubrzava upotrebu strojnog zapažanja u komercijalnim proizvodima. Biblioteka nudi više od 2500 optimiziranih algoritama, uključujući sveobuhvatni skup klasičnih i suvremenih algoritama računalnog vida i strojnog učenja. Ti se algoritmi mogu koristiti za detekciju i prepoznavanje lica, detekciju objekata, klasifikaciju ljudskih radnji u videozapisima, praćenje pokreta kamere, praćenje objekata u pokretu, izvlačenje 3D modela objekata, generiranje 3D oblaka točaka iz stereo kamera, spajanje slika i još mnogo toga. *OpenCV* biblioteka se široko koristi u tvrtkama, istraživačkim skupinama i drugima [11].

NumPy je projekt otvorenog koda s bibliotekom koja omogućava numeričko računanje unutar *Python* programskog jezika. Razvijen je 2005. godine i nastavlja se na prethodne radove

numeričkih biblioteka i biblioteka za rad s poljima. *NumPy* je razvijen i dostupan putem *GitHub*-a, vođen konsenzusom zajednice *NumPy*-a i šire znanstvene zajednice *Python*-a [12].

Imutils je biblioteka koja pruža skup praktičnih funkcija za pojednostavljivanje osnovnih zadataka obrade slika, poput translacije, rotacije, promjene veličine slike, prikaza slika s *Matplotlib*-om, sortiranja kontura, detekcije rubova i još mnogo toga [13].

PIL omogućuje obradu slika unutar *Python* interpretera. Biblioteka pruža podršku za različite formate slika, učinkovito predstavljanje slika i snažne mogućnosti obrade slika. Namijenjena je brzom pristupu podacima pohranjenim u različitim formatima boja te pruža čvrstu osnovu za alate za obradu slika [14].

3.2.3. VisionSDK

VisionSDK je programsko razvojno okruženje (engl. *Software Development Kit* - SDK) kao skup alata i resursa koji pomažu u razvoju aplikacija. Koristi se za implementaciju algoritama i *UseCase*-ova na Alpha ADAS ploči. Ovo više-procesorsko okruženje omogućuje stvaranje različitih tokova podataka za obradu, analizu i prikaz video sadržaja. *VisionSDK* podržava istovremeno izvršavanje na različitim procesorskim jedinicama i temelji se na okviru nazvanom „*Links and Chains*“. Ovaj okvir generira strukturu *UseCase*-a, uspostavlja komunikaciju i tok podataka za definirani *UseCase* te pruža korisničko sučelje „*Link API*“ za interakciju s okvirom.

Links and Chains okvir se koristi za definiranje *UseCase*-ova unutar *VisionSDK*-a. Svaki *Link* se izvršava kao zasebna nit i ima ulazne i/ili izlazne konekcije te vlastitu komunikaciju s drugim *Linkovima*. Svaki *Link* jednoznačno definira algoritam, hardversku vezu na ploči i obradu signala. *Linkovi* se povezuju u *Chain*, pri čemu broj izlaznih konekcija i tip spremnika jednog *Linka* moraju odgovarati broju ulaznih konekcija i tipu spremnika sljedećeg *Linka*. Ova arhitektura olakšava stvaranje *UseCase*-ova i ubrzava razvoj algoritama. *VisionSDK* također pruža *UseCase* generator koji automatski generira potrebne datoteke za rad *UseCase*-a, skraćujući vrijeme potrebno za razvoj. Generator također detektira i izvještava o pogreškama tijekom razvoja *UseCase*-a te omogućuje jednostavniju promjenu toka podataka i veza.

3.3. Razvoj vlastitog programskog rješenja za dobivanje pogleda odozgo na vozilo

U ovome je potpoglavlju detaljnije objašnjen svaki korak razvoja vlastitog rješenja za dobivanje pogleda odozgo na vozilo, pri čemu se opisuju korištene funkcije, formati boja i potrebne matematičke formule pojedinih algoritama.

3.3.1. Snimanje okoline vozila pomoću kamera ADAS ALPHA razvojne ploče i specijaliziranih kamera

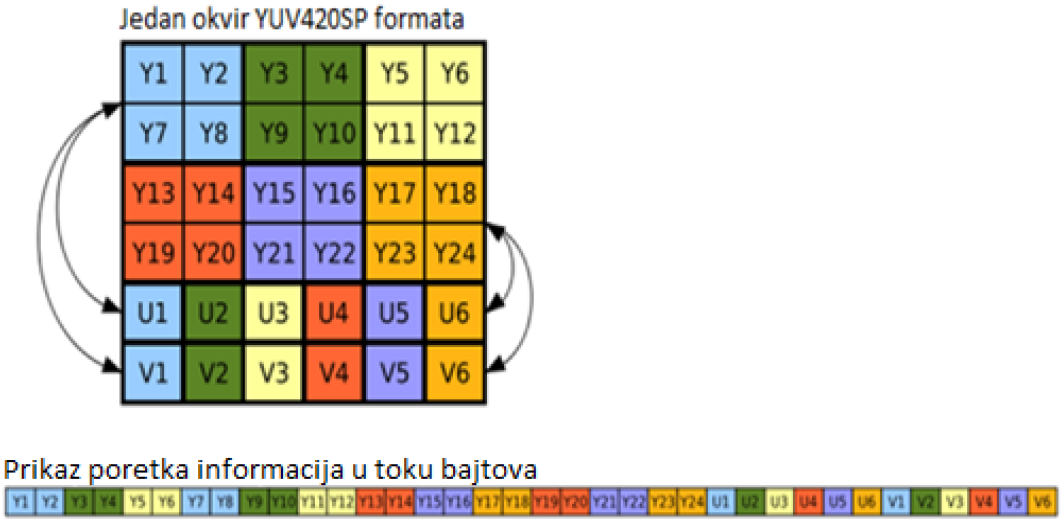
Na maketi autića postavljene su i fiksirane četiri RDACM24B-01 širokokutne kamere [19]. RDACM24B-01 ima OV10640 slikovni senzor i podržava YUV i RAW format s maksimalnom rezolucijom 1280 x 1080 elemenata slike. Pri maksimalnoj rezoluciji moguće je ostvariti maksimalno 30 FPS. RDACM24B-01 ima zakrivljenu *fishe-eye* leću te omogućuje prikaz širokog područja od 180°. Prikaz korištene širokokutne kamere [19] je na slici 3.3.

Slika 3.3. Kamera RDACM24B-01 [19] korištena za snimanje okoline

Formati zapisa koji su korišteni u ovom radu su YUV420SP te YUYV422I. Teorijska osnova ovih formata leži u činjenici da ljudsko oko ima veću osjetljivost na svjetlinu nego na boju. Zbog toga se koristi poduzorak boje kako bi se smanjila količina podataka potrebnih za prijenos i obradu slika, bez značajnog gubitka percepcije boje.

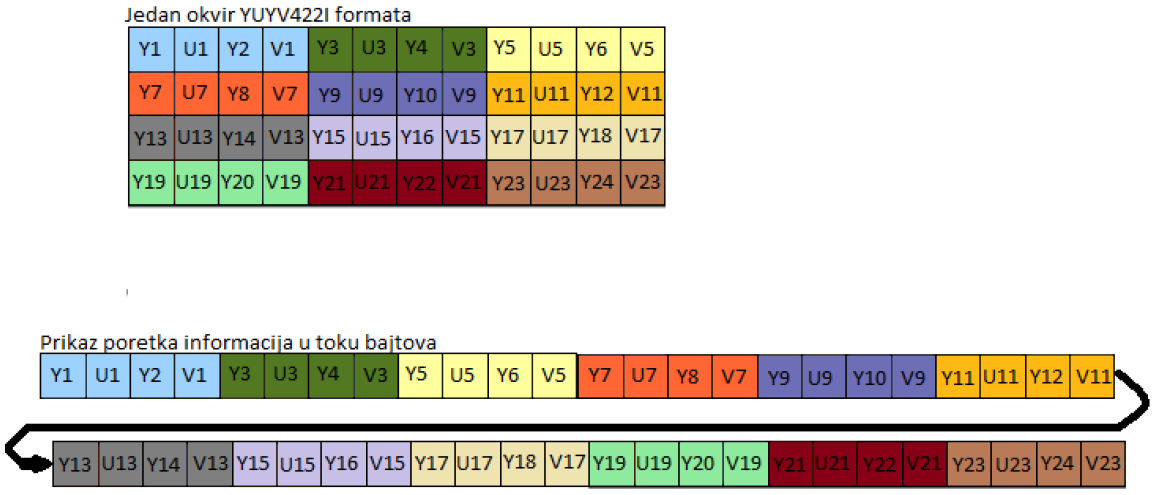
YUV420SP (također poznat kao NV21 ili YVU420SP) je format zapisa koji koristi poduzorak boje, što znači da se boje uzorkuju na manjem broju elemenata slike nego što je prisutno u svakom pojedinačnom elementu slike. U ovom formatu, Y komponenta predstavlja

svjetlinu elemenata slike, dok se U i V komponente koriste za reprezentaciju boje. Prikaz sheme poduzorkovanja te prikaz poretka tih informacija u toku bajtova je na slici 3.4. [10] u nastavku.



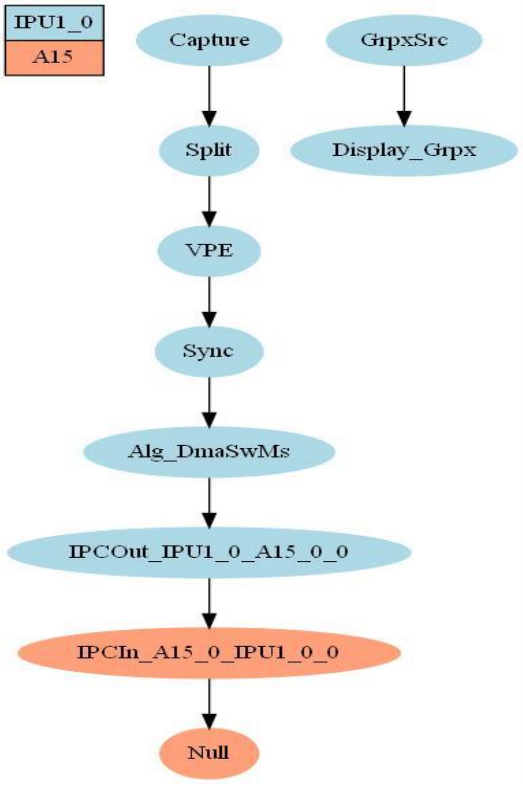
Slika 3.4. Shema poduzorkovanja boja YUV420SP formata zapisa [10]

YUYV422I (također poznat kao YUY2) je format zapisa u kojem svaka dva susjedna elementa slike u jednom retku slike dijele zajedničke informacije o boji, dok svaki element slike ima vlastite informacije o svjetlini. Ova shema kompresije boje omogućuje visoku kvalitetu slike uz manju kompresiju boje u usporedbi s YUV420SP formatom. Prikaz sheme pozicija Y, U, V prikazan je slikom 3.5. Vidljivo je kako YUYV422I format koristi manje poduzorkovanje boja u odnosu na YUV420SP te za istu količinu elemenata slike ima duplo više uzoraka boje što rezultira većim zauzimanjem memorije te zahtjevnijom obradom.



Slika 3.5. Shema poduzorkovanja boja YUYV422I formata zapisa

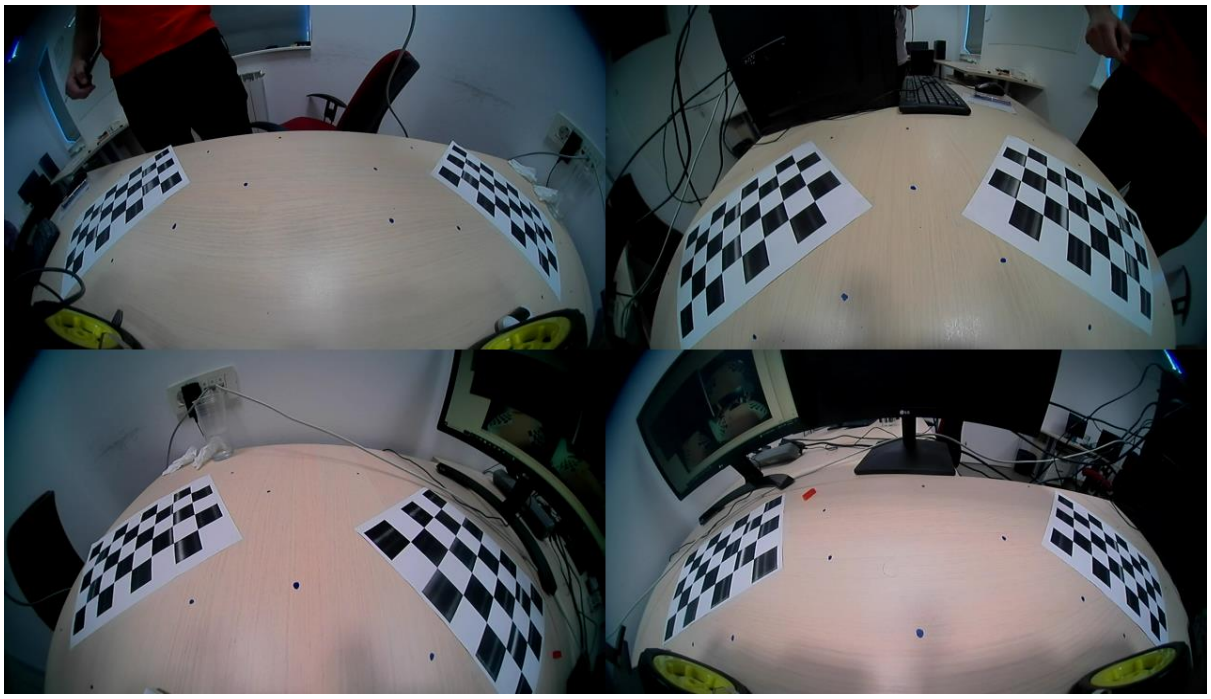
Osnova za razvoj algoritma za stvaranje pogleda odozgo na vozilo je ostvarivanje sinkroniziranog prikaza svih četiriju kamera istovremeno. Pomoću dostupne dokumentacije Alpha ADAS ploče proučeni su izrada, programiranje i uređivanje datoteka razvojnog okruženja. Izrađen je vlastiti *UseCase* kojim je omogućeno snimanje video uradaka četiriju kamera fiksiranih na maketu autića pod horizontalnim kutom od 45°. Omogućen je prikaz na povezanom zaslonu i računalu preko serijskog priključka i *Tera Term* programa. Ploča i računalo na kojemu se prikazuju video snimke moraju biti povezani na istu internetsku mrežu kako bi međusobna komunikacija bila moguća. Sama ploča je povezana s Internetom pomoću Ethernet kabla i odgovarajućeg Ethernet priključka na ploči. Potrebno je koristiti *Capture* (SC) *link* kojim se omogućuje snimanje s većim brojem kamera, u ovom slučaju maksimalno šest kamera koje su fiksirane na maketu autića na SC *SoC*-u. Sljedeći korišteni *link* je *Split link* kojim se ulaz dijeli na više kanala za obradu. Potrebno je dodati vlastiti algoritam koji će ulazne snimke prikazati u mozaik 2x2 pogled i na kraju se postavlja *Null link* koji šalje video na *Ethernet* povezan s računalom. *Null link* radi na A15 procesoru na ploči koji se obično koristi pri korištenju mrežne funkcionalnosti. Prikaz generiranog prikaza *UseCase*-a je na slici 3.6.



Slika 3.6. Dijagram toka generiranog *UseCase*-a za prikaz četiriju slika na razvojnom računalu

Iz prikazanog dijagrama toka generiranog *UseCase*-a vidljivi su korišteni procesori za izvođenje pojedinih dijelova. Tako je proces snimanja okoline te obrade slike odrađen na procesoru IPU1_0 dok je proces prenošenja prikaza preko internetske veze obrađen na A15 procesoru razvojne ploče.

Nakon što je *UseCase* uspješno izrađen te algoritam uspješno prikazuje sve četiri snimke s kamera istovremeno, snimke su prikazane na računalu pomoću odgovarajućih naredbi u *Command Promptu* kojim se snimaju snimke u datoteku na računalu u .bin formatu. Kako bi se video uradak pokrenuo, koristi se *YuvPlayer* jer kamere snimaju u YUV formatu rezolucije 1280 x 720 elemenata slike. Izdvojene su slike iz video uradaka koje se koriste za daljnju obradu kako bi se omogućilo ispravljanje i spajanje prikaza s kamera. Snimljena je okolina vozila s postavljenim uzorcima šahovske ploče kako bi se lakše prepoznale određene točke prikaza, što je potrebno za kalibraciju i ispravljanje slika. Takve slike potrebne su za proces kalibracije slike koji je sljedeći korak u izradi rješenja. Prikaz dobiven izrađenim *UseCase*-om dan je na slici 3.7.



Slika 3.7. Prikaz okoline snimljen četirima kamerama na razvojnoj Alpha ADAS ploči

Iz dobivenog prikaza okoline snimljene kamerama fiksiranim na ploči vidljivo je kako je obuhvaćeno svih 360° oko makete autića. Prikazi su izobličeni zbog načina snimanja širokokutne leće i prikazuju 8 bitni prikaz s 256 vrijednosti svjetline i boja.

3.3.2. Postupak kalibracije kamera

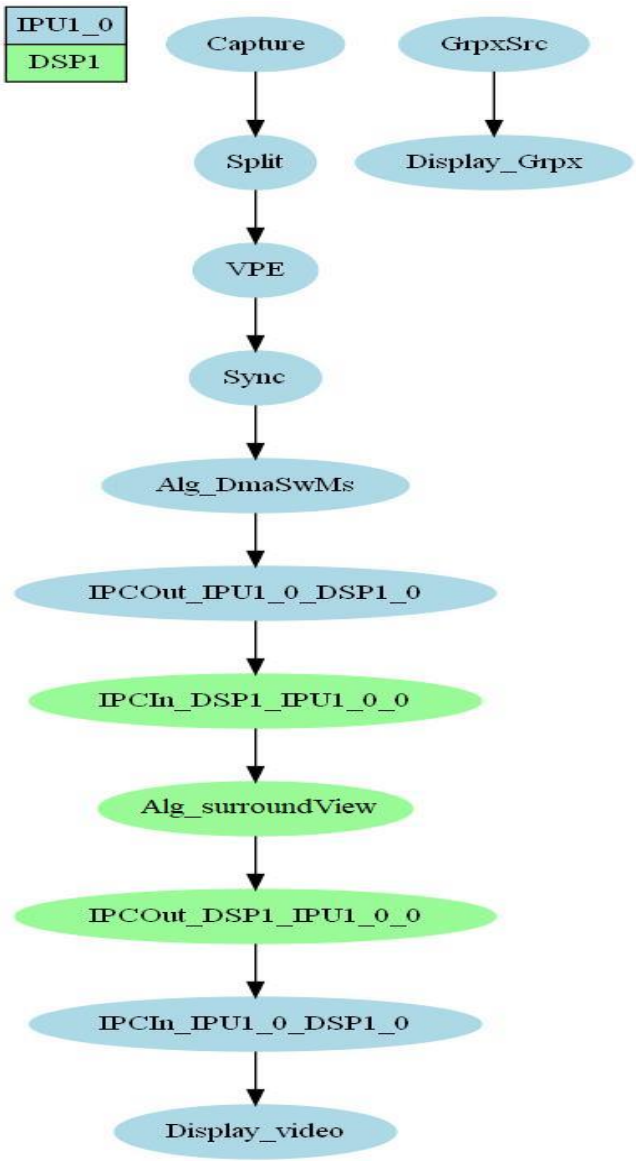
Potrebno je izvesti proces kalibracije svih četiriju kamera fiksiranih na dostupnu maketu autića, kako bi se dobili potrebni parametri za ispravljanje izobličenja širokokutnih kamera. Zbog tzv. *fish-eye* leća širokokutnih kamera dolazi do zakrivljenja pri rubovima snimke s kamere koja snima široko područje od 180°. Postupak je započeo proučavanjem dostupnih *OpenCV* rješenja te programskih jezika u kojima se mogu implementirati. Rješenje kalibracije na osobnom računalu izvedeno je u *Visual Studio Code*-u u programskom jeziku *Python* te korištenjem *OpenCV* biblioteke verzije 4.2.0. koju je potrebno instalirati na računalo. Koristi se *OpenCV* paket `cv2.fisheye` koji omogućuje kalibraciju širokokutnih kamera s *fish-eye* efektom pomoću ranije snimljenih video uradaka i slika na kojima su vidljivi uzorci šahovske ploče pod različitim kutovima i na različitim položajima. Izrađena je *Python* skripta *calibrate.py* u koju je učitan *OpenCV*. Napisan je program koji pronalazi točke na rubovima šahovskih kvadratića te prepoznaje zakrivljenja slike nastala zbog snimanja širokog kuta. Dobiju se ranije spomenuti unutarnji parametri kamere zapisani kao 3x3 matrica **K** i vektor koeficijenata izobličenja *D* koji se koriste za daljnje ispravljanje slika i prikazani su slikom 3.8. u *Python* programskom jeziku zapisani kao *NumPy* polja. Parametri se koriste za funkciju `cv2.fisheye.initUndistortRectifyMap()` kojom se dobiju nove pozicije elemenata slike iz ulazne, neispravljene slike.

Linija ***Kod***

```
1:   K=np.array([[340.21338679435, 0.0, 641.7562987367],
2:                [0.0,342.3529943412, 400.9421658728],
3:                [0.0, 0.0, 1.0]])
4:   D=np.array([[0.004896971589850257],
5:                [0.03606696428275784],
6:                [0.012710154461143392],
7:                [-0.009492837827059527]])
```

Slika 3.8. Prikaz dobivenih parametara za kalibraciju i ispravljanje slike

Dobivenim parametrima omogućena je kalibracija širokokutnih kamera fiksiranih na maketu autića. Parametri se na ploču zapisuju u *C* programskom jeziku, na način da se generira novi *UseCase* koji se nadovezuje na prethodno izrađen za 2x2 mozaik prikaz s četiriju kamera. Izrađuje se algoritam koji se povezuje s prethodnim *Link*-ovima te se u koraku kalibracije kamere u isti algoritam upisuju dobivene pozicije elemenata slike koji u kasnijim koracima služe za ispravljanje izobličenja slika. Prikaz generiranog *UseCase*-a prikazan je slikom 3.9.



Slika 3.9. Dijagram toka generiranog *UseCase*-a za prikaz četiriju slika na zaslonu povezanom s razvojnom pločom

Iz prikazanog dijagrama toka generiranog *UseCase*-a vidljivi su korišteni procesori za izvođenje pojedinih dijelova. Tako je proces snimanja okoline, spajanja prikaza s pojedinih kamera te prikaz slike na zaslonu odrađen na procesoru IPU1_0, dok je proces izvršavanja izvedenog algoritma izvršen na DSP1 procesoru koji je prikladniji za obradu slika te ima puno veću dostupnu memoriju za rad.

3.3.3. Algoritam za ispravljanje izobličenja slika

Snimke snimljene širokokutnim kamerama imaju značajno izobličenje. Izobličenje elemenata slike raste prema rubovima. Da bi se ispravila izobličenja, *OpenCV* omogućava ispravljanje slike funkcijom `cv2.remap()` koja kao parametre prima informacije o slici i parametre dobivene u postupku kalibracije kamere. Vrši se transformacija elemenata ulazne slike i dobije se izlazna slika koja je ispravljena. U tom procesu značajan dio izvorne slike bude izrezan, oko 30% slike kao što je prikazano slikom 3.12. u nastavku. Kako bi se vratilo izgubljene elemente slike, potrebno je urediti skriptu kako bi se zadržala izvorna veličina slike. Uvedeno je dodatno skaliranje slike tako što je zadana dimenzija slike koja se želi zadržati nakon ispravljanja. *OpenCV* nudi mogućnost unosa `balance` vrijednosti koja može primiti vrijednosti od 0 do 1. Ukoliko je ta vrijednost 0, zadržava se samo dio slike koji je izvorne kvalitete bez većeg gubitka elemenata slike i potrebne interpolacije, što znači da se dio slike pri rubovima izreže. Ukoliko je ta vrijednost 1, ispravljena slika se skalira na veličinu ulazne slike te se dobije nova vrijednost matrice **K** koja se predaje funkciji `cv2.fisheye.initUndistortRectifyMap()`. Navedena funkcija omogućava mapiranje ulaznih elemenata slike kako bi veličina izlazne, ispravljene slike ostala ista kao ulazna. Mapirani elementi se zatim predaju funkciji `cv2.remap()` koja mijenja vrijednosti elemenata ulazne slike u elemente ispravljene slike. Kako dolazi do gubitka elemenata slike prilikom ispravljanja, potrebno je izvesti interpolaciju koja se definira kao parametar funkcije `cv2.remap()` te dolazi do manjeg gubitka i smanjenje kvalitete slike. Funkcija `cv2.remap()` vraća vrijednosti x' i y' koordinata elemenata ispravljene slike koje se spremaju u datoteku potrebnu za implementaciju ispravljanja slika na razvojnom okruženju kao što je prikazano slikom 3.10.

Linija ***Kod***

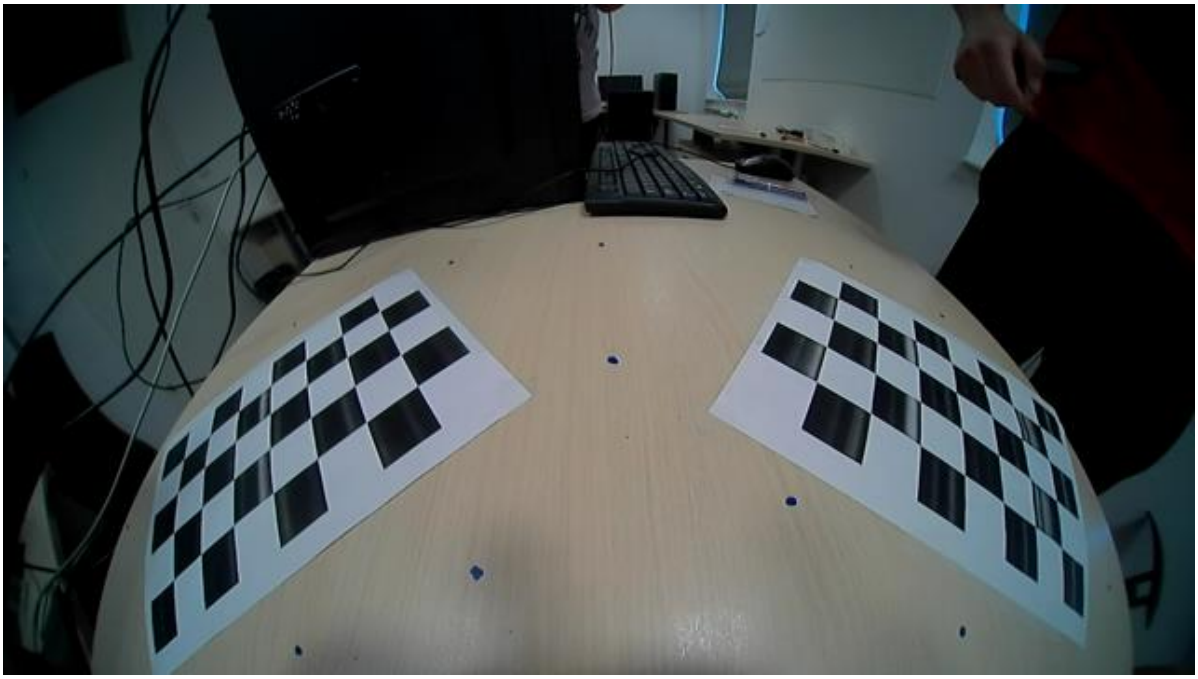
```
1:   map1,map2=cv2.fisheye.initUndistortRectifyMap(scaled_K,
2:                  D, np.eye(3), new_K,  dim3,  cv2.CV_16SC2)
3:   undistorted_img = cv2.remap(img, map1, map2,
                                 interpolation=cv2.INTER_LINEAR,
4:                               borderMode=cv2.BORDER.CONSTANT)
5:   content = map1
6:   file.write(str(map1))
```

Slika 3.10. Prikaz dijela koda za kalibraciju i ispravljanje slike u *Python* programskom jeziku

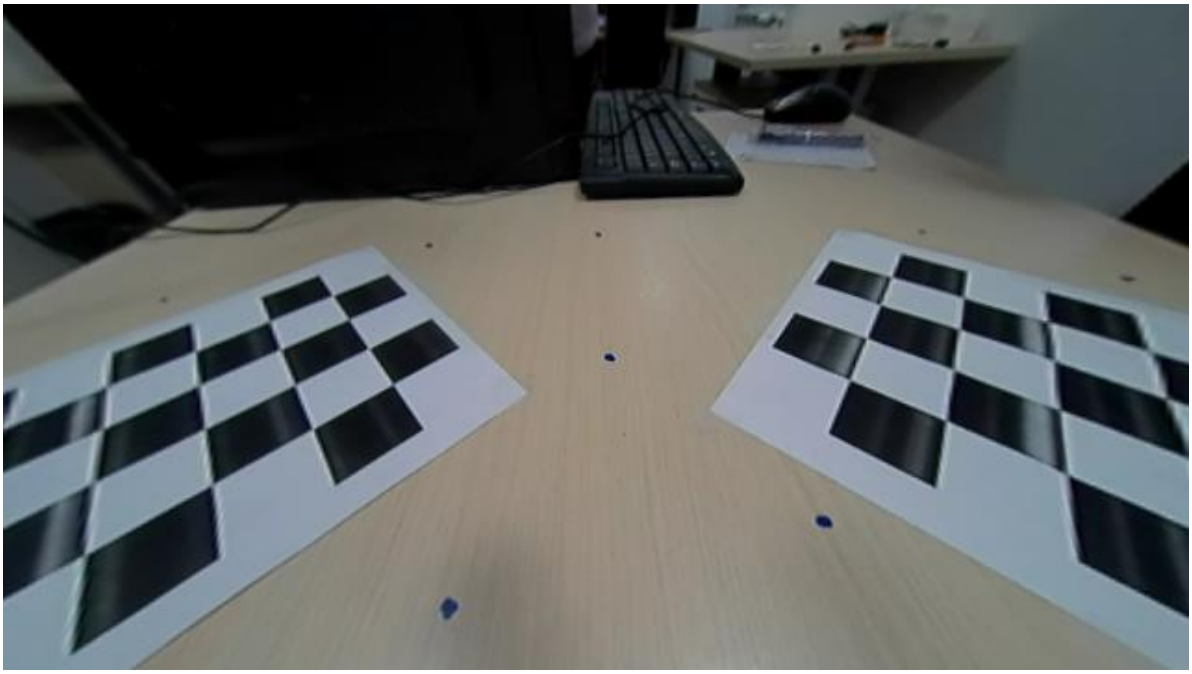
Nove koordinate su prepisane u *C* programski jezik u obliku polja cjelobrojnih vrijednosti. Zbog YUV formata zapisa kamera i ograničenosti memorije na samoj ploči, rješenje je uspješno implementirano uz određene gubitke na kvaliteti. Omogućen je istovremeni prikaz svih četiriju slika s kamera te je izrađen vlastiti algoritam koji ispravlja slike. Nakon toga potrebno je izvesti transformaciju perspektive i spajanje susjednih slika na rubovima gdje se snimke poklapaju svojim prikazom.

U nastavku, slika 3.11. predstavlja prikaz s prednje kamere prije ispravljanja izobličenja, slika 3.12. predstavlja prikaz s prednje kamere nakon ispravljanja izobličenja na osobnom računalu bez skaliranja, dok slika 3.13. predstavlja prikaz nakon ispravljanja izobličenja na osobnom računalu s dodatnim skaliranjem. Konačno, slika 3.14. prikazuje rezultat ispravljanja prikaza s četiriju kamera s dodatnim skaliranjem na razvojnoj ploči.

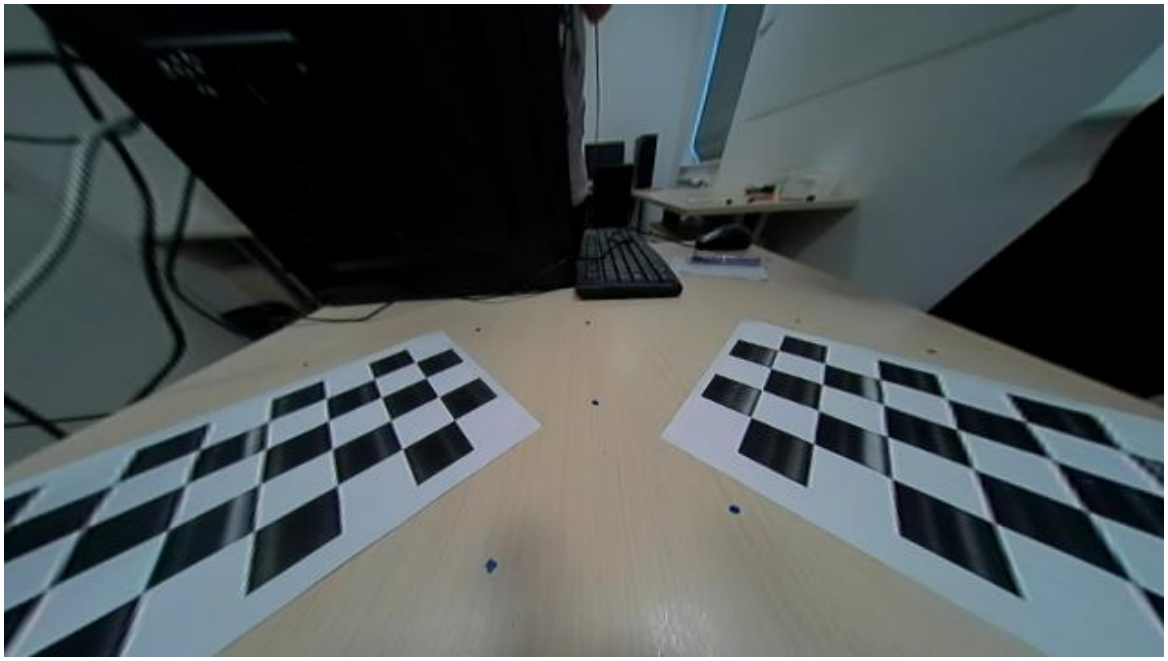
Na prikazu izvedenog ispravljanja izobličenja slika s kamera vidljivo je kako su svi zakrivljeni rubovi uspješno ispravljeni. Izobličenja su, kao što je već spomenuto, veća pri rubovima te prilikom ispravljanja dolazi do većeg gubitka informacija i gubitka detalja, tako da su prikazi pri rubovima manje kvalitete. Isto tako dolazi do širenja objekata na rubovima, što znači da objekti koji se nalaze na rubu slike izgledaju malo šire nego što to jesu u stvarnosti. Rezultat ispravljanja izobličenja slika je prikaz okoline sličnije stvarnoj.



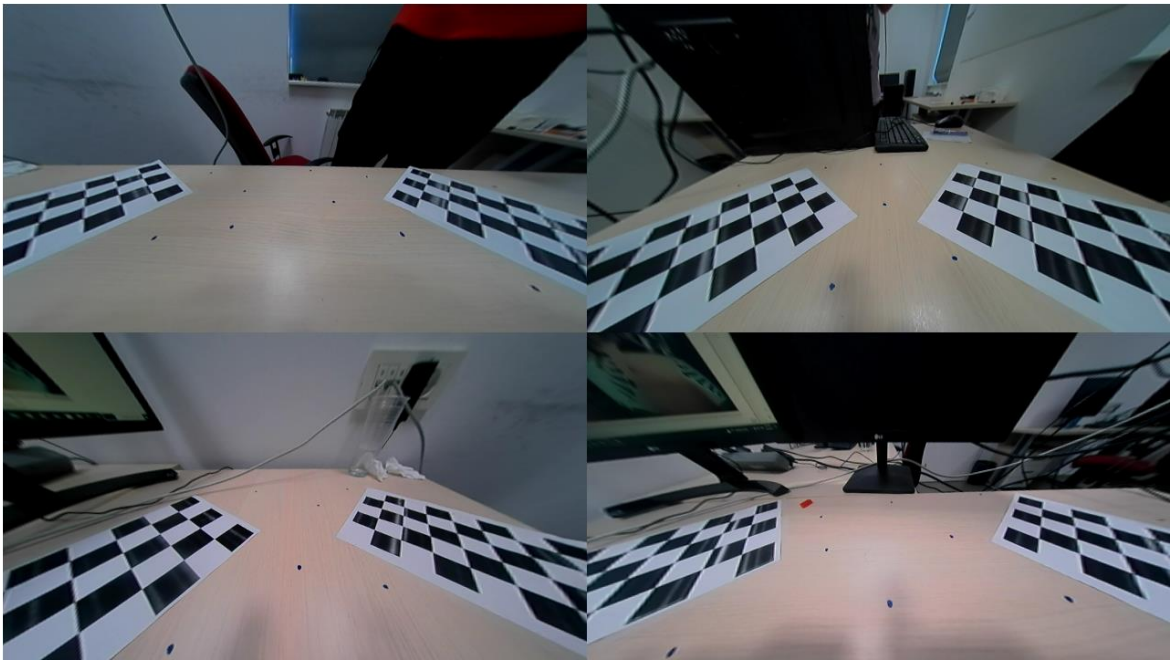
Slika 3.12. Izvorni prikaz slike s prednje kamere razvojnog okruženja



Slika 3.11. Prikaz slike s prednje kamere nakon ispravljanja izobličenja bez dodatnog skaliranja



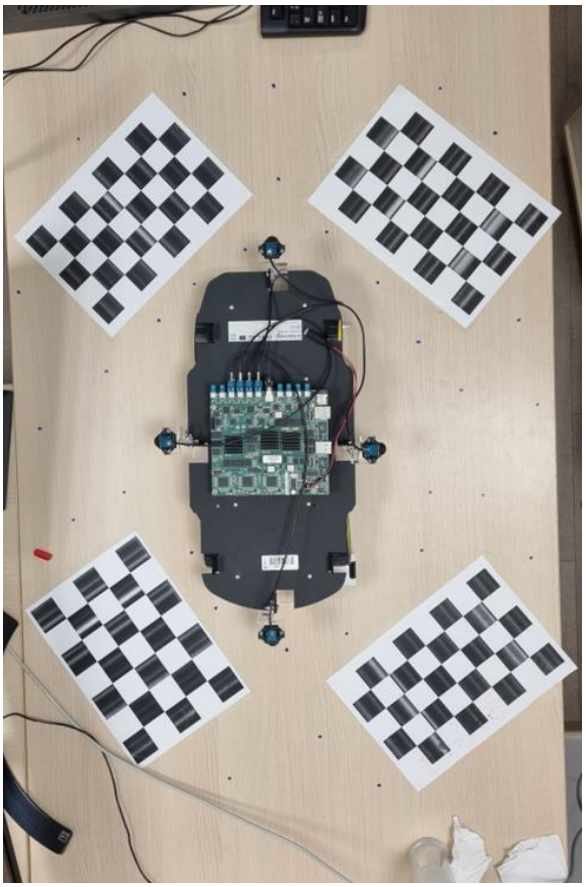
Slika 3.14. Prikaz slike s prednje kamere nakon ispravljanja izobličenja sa skaliranjem



Slika 3.13. Prikaz okoline snimljen kamerama na razvojnoj ploči nakon ispravljanja izobličenja

3.3.4. Algoritam za prebacivanje perspektive slike u ptičju perspektivu

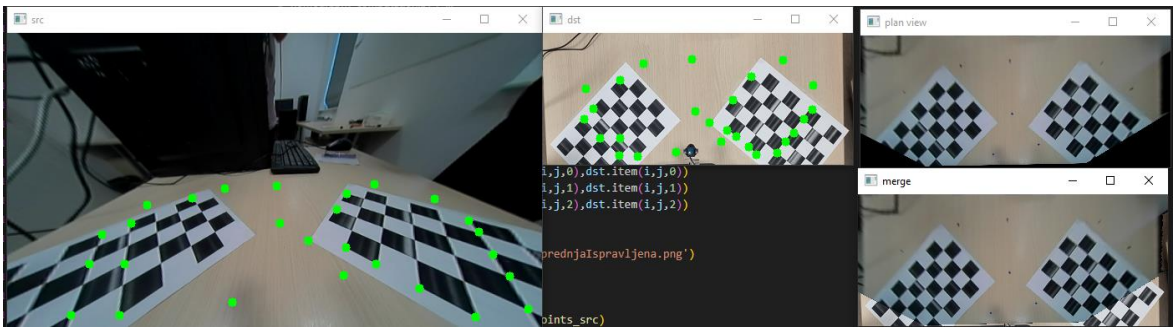
Nakon što su slike ispravljene, potrebno je odrediti i iskoristiti matricu homografije kojom se slika zakreće i mijenja perspektiva same slike na pogled odozgo. Kako bi se dobila matrica homografije, potrebno je snimiti referentnu sliku koja prikazuje okolinu vozila snimljenu pod pravim kutom odozgo. Referentna slika predstavlja željeni prikaz kojemu se teži prilikom izrade algoritma i prikazana je slikom 3.15.



Slika 3.15. Referentni prikaz makete autića i okoline dobiven fotografiranjem odozgo pomoću kamere mobilnog uređaja

Nakon što je dobivena referentna slika, pomoću nje se transformira perspektiva prikaza snimljenog kamerama makete autića te se pomoću *OpenCV* funkcije `cv.findHomography()` dobije odgovarajuća matrica homografije za željenu promjenu perspektive. Za dobivanje matrice homografije prikaza izrađena je interaktivna *Python* skripta koja omogućuje učitavanje po dvije slike, referentne i slike koja se obrađuje. Omogućeno je ručno označavanje ključnih točaka koje predstavljaju iste točke prostora na slikama te pomoću

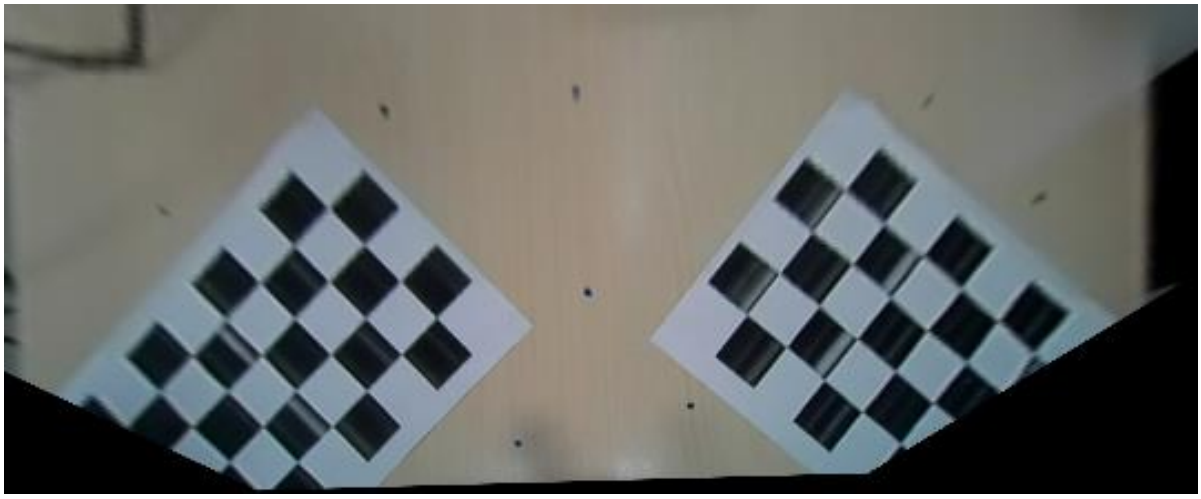
minimalno četiriju parova ključnih točaka skripta omogućava računanje željene matrice homografije koja je detaljnije objašnjena u dijelu 2.1.3. Prikaz procesa dobivanja matrice homografije označavanjem ključnih točaka na referentnoj slici i slici na koju se primjenjuje matrica prikazan je slikom 3.16. Na slici 3.16. lijevi prozor prikazuje ulaznu sliku na koju se primjenjuje matrica homografije, srednji prozor prikazuje referentnu sliku, dok desni gornji prozor prikazuje ulaznu sliku s transformiranom perspektivom. Posljednji prozor prikazuje spajanje dijelova slike s transformiranom perspektivom i odgovarajućih dijelova referentne slike, kako bi se postigao cjelovit i koherentan rezultat koji daje uvid u točnost transformacije perspektive.



Slika 3.16. Prikaz procesa transformacije perspektive označavanjem ključnih točaka

Zelenim točkama označene su prethodno spomenute ključne točke. Izrađena skripta omogućuje ispis matrice homografije, dobivenog pogleda nakon transformacije te prikaz spojene referentne i ulazne slike nakon transformacije perspektive.

Kako bi se primijenila matrica homografije na ulaznu sliku, koristi se funkcija `cv.warpPerspective(src, M, dsize)` koja kao argumente prima ulaznu sliku označenu sa `src`, dobivenu matricu homografije označenu s `M`, željenu rezoluciju slike označenu s `dsize` te vraća novu sliku s promijenjenom perspektivom. U nastavku slikom 3.17. prikazana je slika nakon promjene perspektive. Postupak transformacije perspektive zahtijeva promjenu pozicija elemenata slike kako bi se simulirao ptičji pogled. Kada se perspektiva slike snimljene pod kutom od 45° prebaci u ptičju perspektivu, dolazi do izobličenja i promjene oblika slike. U tom postupku nastaju prazni dijelovi slike. Na primjeru dobivene slike nakon promjene perspektive to su crni dijelovi nastali zbog rezanja i proširivanja slike kako bi se uklopila u novi perspektivni pogled.



Slika 3.17. Prikaz s prednje kamere nakon transformacije perspektive u pogled odozgo

3.3.5. Implementacija prebacivanja perspektive slike u ptičju perspektivu na razvojnu ploču

Nakon što je rješenje razvijeno na računalo, bilo je potrebno implementirati rješenje na razvojno okruženje odnosno Alpha ADAS ploču. Uz dobivene matrice homografije za sve četiri slike s kamera potrebno je prepisati ih u *C* programski jezik za ploču te prepisati funkciju `cv.warpPerspective()` za svaku kameru kako bi se promijenila perspektiva slika. Funkcija je uspješno implementirana na razvojnu ploču u *C* programskom jeziku, ali zbog ograničenosti memorijom i povećanja vremena izvođenja algoritma prilikom računanja i rada s decimalnim vrijednostima, implementirane su nove pozicije elemenata slike prilikom transformacije u obliku polja cjelobrojnih vrijednosti.

U nastavku slikom 3.18. prikazan je dio koda *Python* skripte u kojemu je iskorištena prethodno dobivena matrica homografije prednje kamere te su po formulama (2-21) i (2-22) izračunate nove vrijednosti x i y koordinata elemenata slike za transformaciju perspektive slike. Nove vrijednosti x i y spremljene su u datoteku koja se učitava u *C* programskom jeziku te se prilagođava za implementaciju na razvojnu ploču.

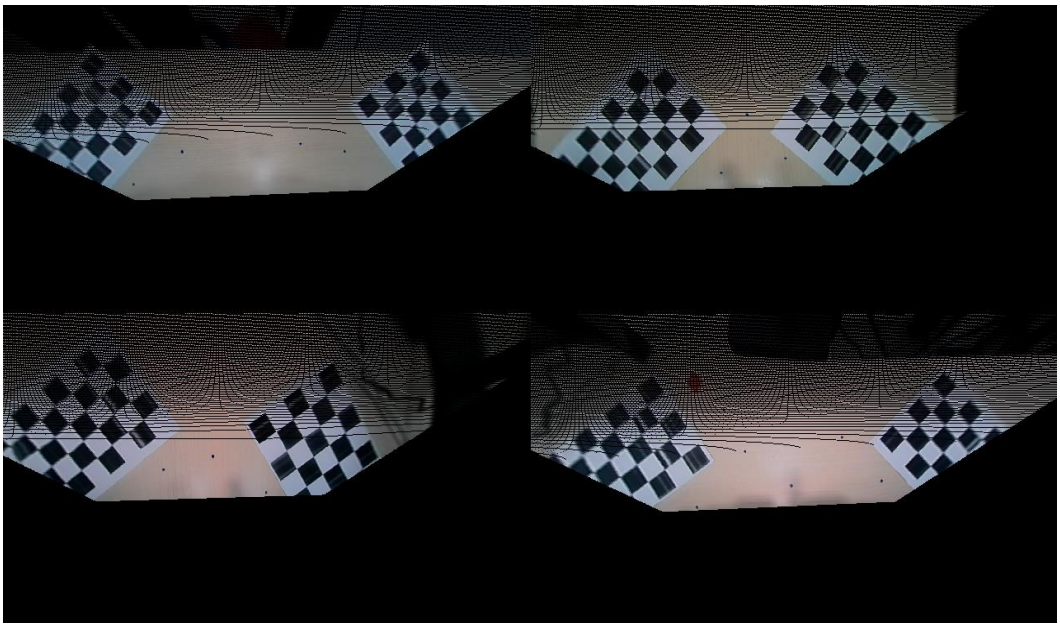
Linija ***Kod***

```
1:   H = [[-1.10304866e+00, -2.31634327e+00,  5.83991518e+02],
2:        [-6.09973803e-02, -2.54551002e+00,  4.38940685e+02],
3:        [-4.18294661e-04, -8.85905986e-03,  1.00000000e+00]]
4:   for y in range(360):
5:     for x in range(640):
6:       x1 = int((H[0][0] * x + H[0][1] * y + H[0][2]) /
7:                (H[2][0] * x + H[2][1] * y + H[2][2]))
8:       y1 = int((H[1][0] * x + H[1][1] * y + H[1][2]) /
9:                (H[2][0] * x + H[2][1] * y + H[2][2]))
10:  map1 = y1, x1
11:  file.write(str(map1))
```

Slika 3.18. Dio koda za primjenu matrice homografije na elemente slike

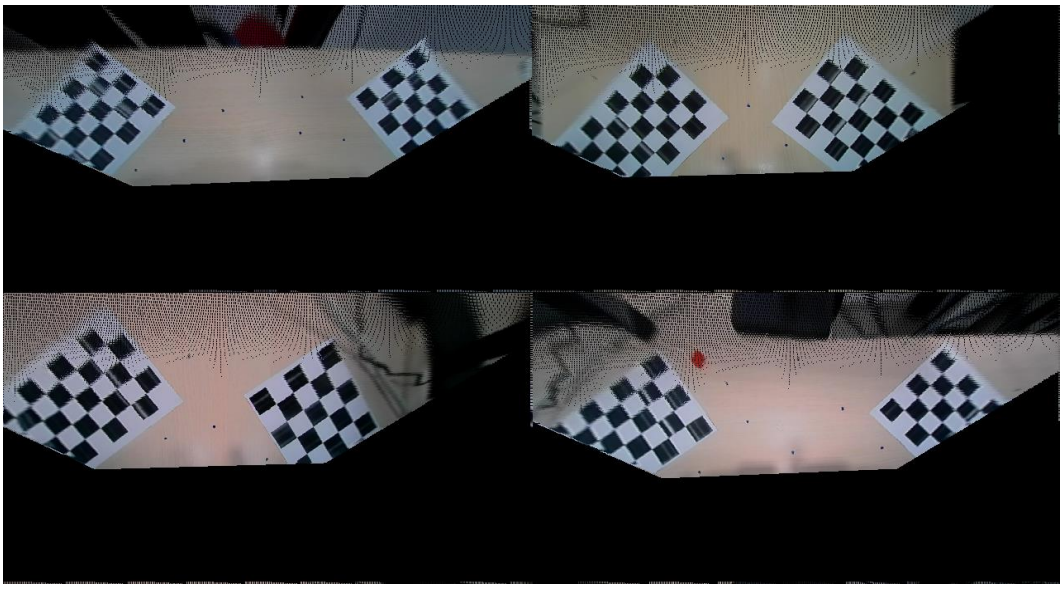
Nove pozicije elemenata slike spremljene su u polja cjelobrojnih vrijednosti. Treba naglasiti da su na ploču, zbog nedostatka memorije, implementirane transformirane vrijednosti pozicija elemenata slike samo za dvije matrice homografije, prednju i lijevu. Teoretski su dvije matrice dovoljne jer bočne kamere i kamere na prednjoj i zadnjoj strani obuhvaćaju jednako područje, ako su na točno određenim pozicijama automobila, u ravnini jedna s drugom. Nedostatak nemogućnosti implementiranja homografija za sve četiri kamere, u ovakvom realnom sustavu gdje kamere nisu savršeno usklađene te svaki milimetar odstupanja od referentne pozicije i kuta snimanja utječe na ishod transformacije, dovodi do nesavršenosti i odstupanja od željenog rezultata.

Algoritmom za transformaciju perspektive se prolazi kroz elemente slike te se premještaju uz primjenu matrice homografije. Rezultat ove transformacije prikaza sa svih kamera na razvojnoj ploči prikazan je slikom 3.19.



Slika 3.20. Prikaz snimljen kamerama razvojnog okruženja nakon promjene perspektive u ptičju perspektivu

Očito je kako je potrebna interpolacija elemenata slike koji su se prilikom promjene perspektive izgubili. Korištena je interpolacija najbližim susjedom (engl. *Nearest Neighbour*) kojom se nalazi vrijednost najbližeg susjednog elementa slike koja se postavlja na „prazni“ element slike. U nastavku slika 3.20. predstavlja prikaz pogleda sa svih četiriju kamera nakon implementirane interpolacije.

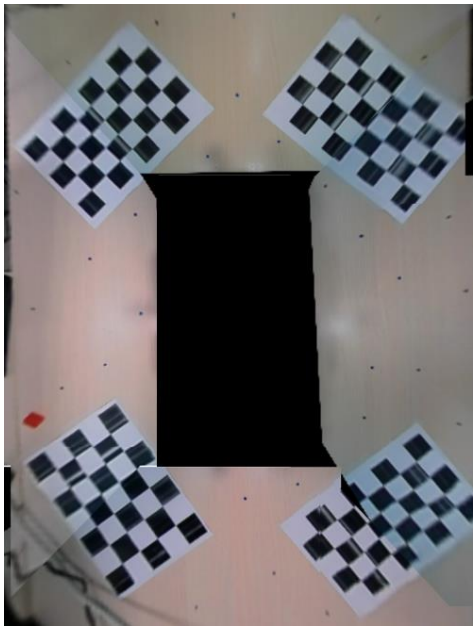


Slika 3.19. Prikaz snimljen kamerama razvojnog okruženja nakon promjene perspektive i primijenjene interpolacije

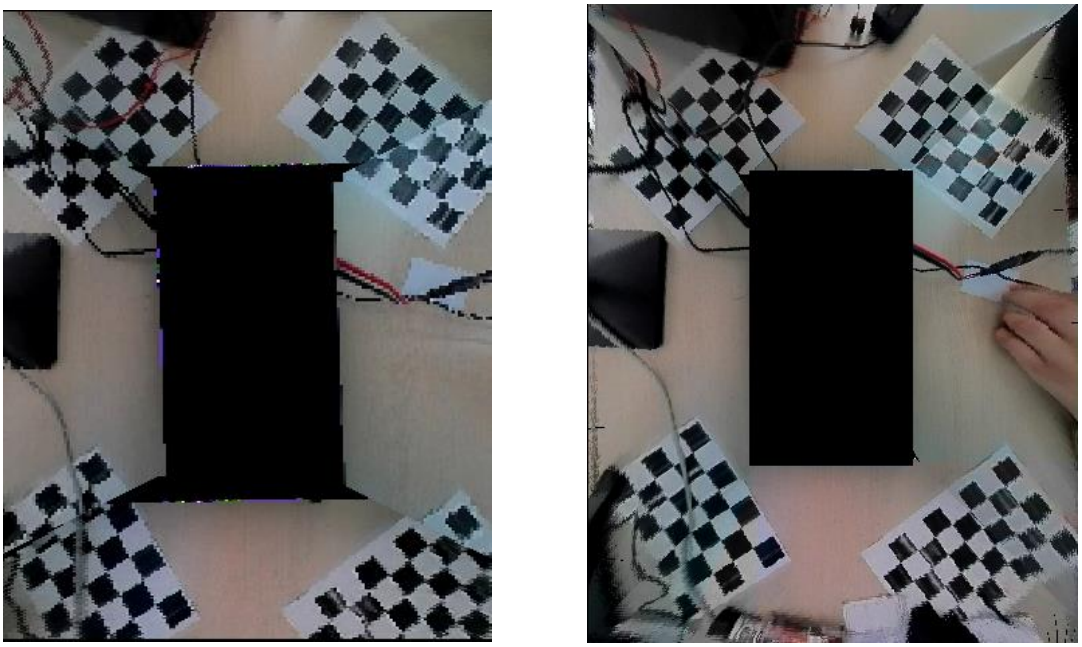
Nakon implementirane interpolacije, vidljivo je prisutno manje odstupanje pri gornjim dijelovima prikaza zbog prevelikog „gubitka“ elemenata slika. Prikaz pogleda odozgo na vozilo kao pomoć pri parkiranju ne zahtjeva prikaz objekata koji su na većoj udaljenosti od automobila tako da se kao „područje interesa“ uzima približno 2 metra oko automobila.

3.3.6. Način spajanja slika u jedinstveni pogled odozgo na vozilo

Nakon što je promijenjena perspektiva prikaza sa svih četiriju kamera, potrebno je izvršiti rotaciju i spajanje susjednih prikaza kamera. Može se odrediti točna pozicija na kojoj dolazi do preklapanja susjednih pogleda jer su kamere fiksirane na maketu autića te ne dolazi do promjene položaja i kuta snimanja kamera. Slike sa svake kamere postavljaju se na određenu lokaciju na konačnoj slici, ovisno o vidnom području kamera. U ovom dijelu ručno su definirane lokacije slika s kamera te su zakrenute pod kutovima ovisno o strani makete autića na kojoj kamere snimaju. Tako je prikaz s kamere na lijevoj strani makete auta zakrenut za 90° u smjeru obrnuto od kazaljke na satu, prikaz s desne kamere 90° u smjeru kazaljke na satu, dok je prikaz s kamere na zadnjoj strani auta zakrenut za 180° u odnosu na smjer snimanja prednje kamere. U nastavku su spojeni pogledi prostora oko automobila izvedeni na razvojnom računalu prikazani slikom 3.21., dok slike 3.22. a) i b) predstavljaju spojene poglede izvedene na razvojnoj ploči s različitim formatima zapisa.



Slika 3.21. Prikaz spojenih pogleda kamera izrađen na razvojnom računalu



a) b)

Slika 3.22. Prikaz spojenih pogleda kamera izrađen na razvojnoj ploči (a) s YUYV422I formatom zapisa (b) s YUV420SP formatom zapisa

Iz dobivenih prikaza izrađenih na razvojnom računalu te razvojnoj ploči vidljivo je određeno odstupanje i razlika u kvaliteti dobivenih prikaza odozgo. Rješenje na razvojnom računalu razvijeno je u rezoluciji 1280 x 720 elemenata slike s formatom zapisa YUYV422. Samim time, rješenje na računalu ima najveću kvalitetu i najbolje spojene rubove prikaza sa susjednih kamera zbog veće procesorske moći te korištenja dostupnih *OpenCV* rješenja koja omogućavaju brže procesiranje i kvalitetnije ispravljanje i transformaciju perspektive s različitim interpolacijama. Rješenje izrađeno s YUYV422I formatom zapisa prikazuje skalirani prikaz okoline vozila. Zbog manjka resursa na razvojnoj ploči, smanjena je rezolucija prikaza s 1280 x 720 na 640 x 360 elemenata slike te uzorkovanje boja i svjetline kako bi rješenje brže radilo te ostvarilo brzinu prikaza od 6.5 FPS. Prilikom skaliranja prikaza došlo je do većih odstupanja, posebno vidljivim pri rubovima na desnoj strani automobila. Jedan od uzroka odstupanja je i nesavršenost izvođenja snimanja okoline prilikom koje kamere moraju biti u točnoj poziciji i pod istim kutom kao prilikom izrade algoritma. Kao što je već ranije spomenuto u dijelu 3.3.5, implemetirana je homografija samo za lijevu kameru, što objašnjava veće odstupanje na desnoj strani automobila. Rješenje s YUV420SP formatom predstavlja rješenje

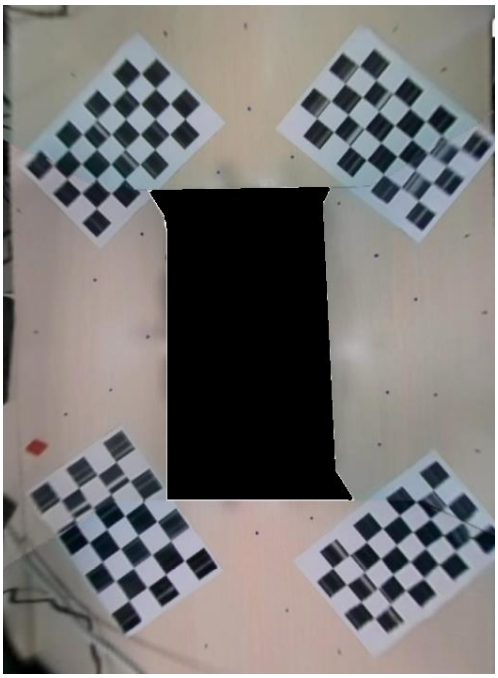
izrađeno s rezolucijom 1280 x 720 elemenata slike, što je omogućeno jačim poduzorkovanjem boje te samim time manjim zauzimanjem memorije prilikom izrade. Rubovi su spojenih s manjim odstupanjima, ali je vrijeme izvođenja ovog programskog rješenja dulje, čime ostvaruje brzinu prikaza od 2.5 FPS.

3.3.7. Algoritam za ujednačavanje osvjetljenja na slikama

Kada se četiri slike dobivene s kamera spoje u jedinstveni pogled odozgo, dolazi do preklapanja područja slika koja pokrivaju isto područje vozila. Da bi se postiglo spajanje slika bez primjetnih prijelaza, potrebno je izvršiti fotometrijsko poravnanje. Postupak fotometrijskog poravnanja može se izvesti koristeći različite tehnike, poput ujednačavanja histograma ili usklađivanja boje. Ove tehnike se primjenjuju na preklapajuća područja slika kako bi se postigla ujednačena svjetlina i boja.

Prilikom fotometrijskog poravnanja susjednih područja koja se preklapaju, vrijednosti elemenata slike se obično množe s faktorom koji se određuje na temelju srednje vrijednosti ili histograma svjetline i boje. Metoda korištena za ujednačavanje osvjetljenja u ovom radu je *gamma* korekcija [18]. *Gamma* korekcija mijenja intenzitet elemenata slike prema određenoj *gamma* funkciji. To omogućuje kontrolu kontrasta i svjetline slike. *Gamma* vrijednost može se prilagođavati za svaki blok slike kako bi se postigao ujednačen izgled između susjednih blokova. Pravilan odabir faktora ili parametra ovisi o specifičnim karakteristikama slika i željenom izgledu između susjednih blokova.

Računanje *gamma* vrijednosti s kojom je potrebno množiti YUV vrijednosti komponenti svakog elementa slike prikazano je u dijelu 2.2.1. S obzirom da se provodi mjerenje na 8-bitnim slikama čije su vrijednosti Y, U, V komponenti u rasponu od 0 do 255, računaju se korekcije množenja u tom rasponu. Dobije se takozvana „tablica pretraživanja“ koja u sebi sadrži 256 elemenata vrijednosti konačne svjetline i boja za izvorne vrijednosti početnih prikaza. Prolaskom kroz svaki element prikaza primjenjuje se „tablica pretraživanja“ te se proračunavaju nove, usklađene vrijednosti svjetline i boje. U nastavku je slikom 3.23. prikazan jedinstveni prikaz dobivenog pogleda odozgo na vozilo nakon što je provedeno fotometrijsko poravnanje spojenih slika pomoću *Python* programskog jezika.



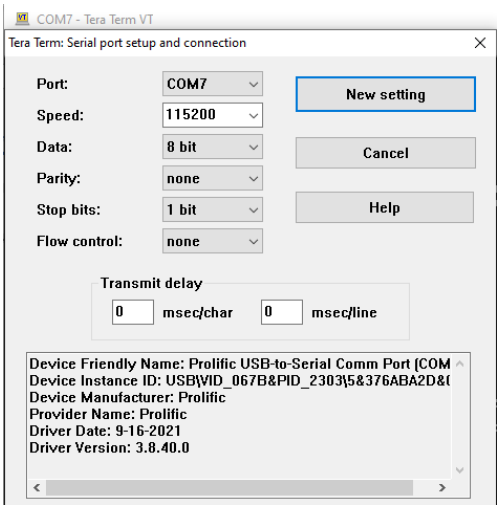
Slika 3.23. Prikaz spojenih pogleda kamera izrađen na razvojnom računalu nakon fotometrijskog poravnanja

Implementirano je fotometrijsko poravnanje prikaza sa svih četiriju kamera te su razlike svjetline i boje susjednih prikaza kamera minimalne i samim time su rubovi bolje spojeni te daju vjerniji jedinstveni prikaz. Slike su postigle značajno ujednačeniji izgled s konzistentnim osvjetljenjem i bojama, čime je poboljšana kvaliteta i vizualni dojam svakog pojedinog prikaza.

Implementacija fotometrijskog poravnanja nije uspješno izvedena na razvojnu ploču jer zahtjeva veću procesorsku moć. Vrši se množenje svih elemenata slike s decimalnim vrijednostima, što zahtjeva dodatno zauzimanje veće količine memorije koja je, zbog zahtjevnosti izvođenja prethodnih koraka razvijenog rješenja, već iskorištena.

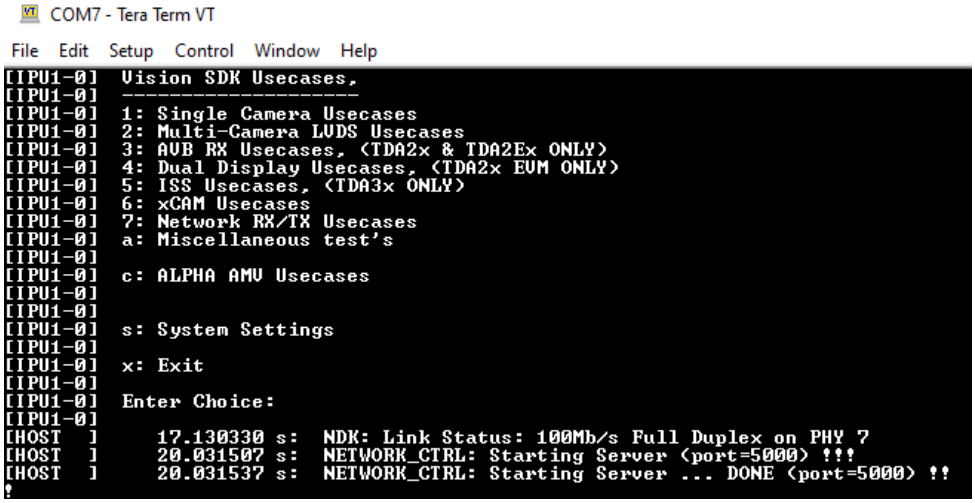
3.3.8. Način pokretanja programskih rješenja na razvojnoj ploči

Programska rješenja na razvojnoj ploči se pokreću pomoću ranije spomenutog *Tera Term* komunikacijskog sučelja. Ploča se UART kabelom povezuje s računalom te se konfigurira priključak na koji je povezana. Postavlja se brzina komunikacije razvojnog računala na brzinu komunikacije s razvojnom pločom koja iznosi 115200 *Baud*-a. Prikaz konfiguriranog priključka s kojim je povezana razvojna ploča prikazan je slikom 3.24.



Slika 3.25. Prikaz konfiguriranog priključka COM7

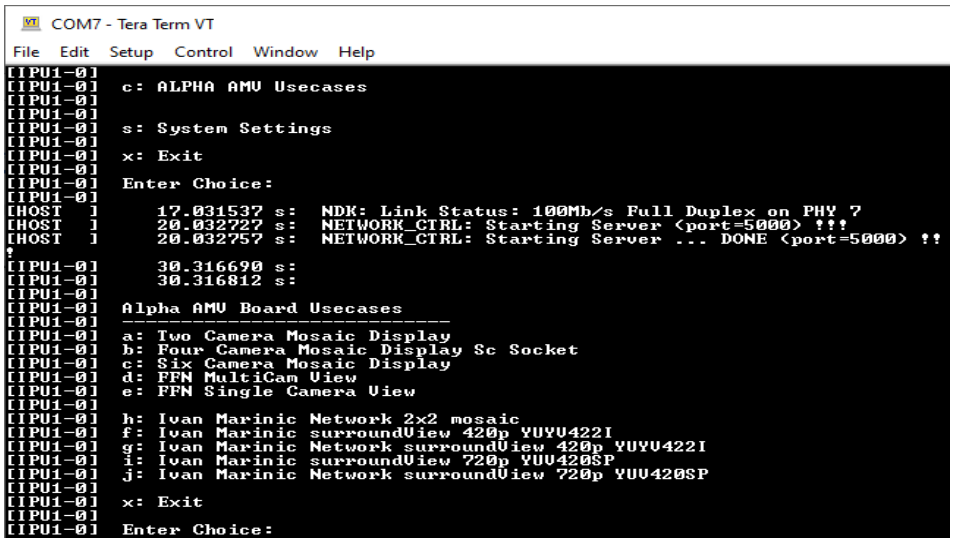
Nakon što je priključak konfiguriran, razvojna ploča se napaja te se otvara početni izbornik za pokretanje implementiranih *UseCase*-ova prikazan slikom 3.25. Kako bi se pokrenuli *UseCase*-ovi razvijeni za korištenu razvojnu ploču, potrebno je pritiskom tipke znaka „c“ razvojnog računala otvoriti izbornik „APLHA AMV *Usecases*“.



Slika 3.24. Prikaz početnog izbornika *Tera Term*

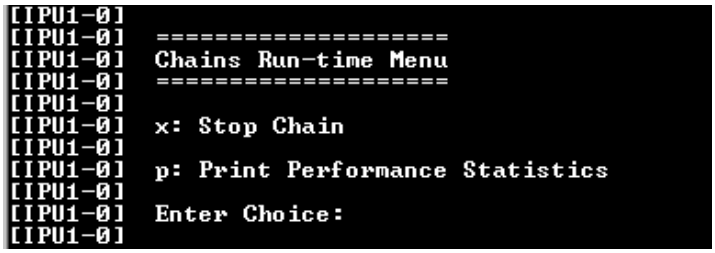
Unutar odabranog izbornika nalaze se razvijeni *UseCase*-ovi razvojne ploče. Odabirom tipke određenog prikazanog znaka pokreće se željeni *UseCase*. U nastavku, slikom 3.26. prikazani su razvijeni *UseCase*-ovi potrebni za pokretanje algoritma za dobivanje pogled odozgo. Odabirom znaka „*h*“ pokreće se prikaz slika sa svih četiriju kamera prikazan u dijelu 3.3.1. Odabirom znaka „*f*“ pokreće se prvo rješenje algoritma s YUYV422I formatom zapisa prikazano na zaslonu, dok znak „*g*“ prikazuje rješenje na razvojnom računalu preko internetske

konekcije. Slijedno tome, znak „*i*“ pokreće razvijeno rješenje s YUV420SP formatom zapisa s prikazom na zaslonu, dok „*j*“ omogućuje prikaz videozapisa na razvojnom računalu.



Slika 3.26. Prikaz izbornika implementiranih *UseCase*-ova razvojne ploče AMV Alpha

Kada je *UseCase* pokrenut, omogućeno je zaustavljanje istog pritiskom znaka „*x*“ na razvojnom računalu. Omogućen je i prikaz statistika performansi tijekom izvođenja *UseCase*-a. Tako je pritiskom na znak „*p*“ omogućeno praćenje i analiza brzine i kvalitete izvođenja samog programskog rješenja. Slika 3.27. prikazuje ispis mogućnosti prilikom izvođenja odabranog rješenja.



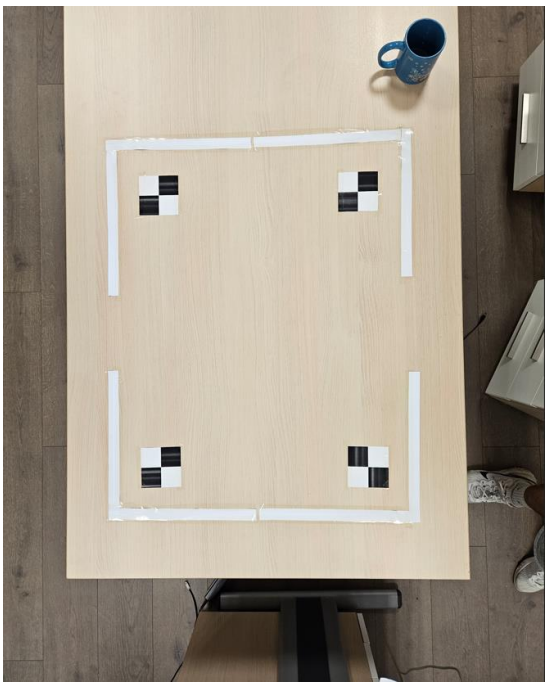
Slika 3.27. Prikaz izbornika s mogućnostima izrađenog rješenja tijekom izvođenja

4. TESTIRANJE PREDLOŽENOG RJEŠENJA ZA DOBIVANJE POGLEDA ODOZGO NA VOZILO

Unutar ovog poglavlja prikazat će se rezultati dobivenih pogleda odozgo na vozilo implementiranih na AMV Alpha ADAS ploču. Prikazat će se rezultati dobiveni s formatom zapisa YUV420SP te YUYV422I i usporediti s referentnim slikama pogleda odozgo. Izmjerit će se vrijeme izvođenja algoritma iskazano preko broja FPS. Tabličnim prikazom bit će prikazano vrijeme izvođenja pojedinih dijelova algoritma te tzv. „gubljenje okvira“ tokom provođenja ispravljanja i promjene perspektive okvira.

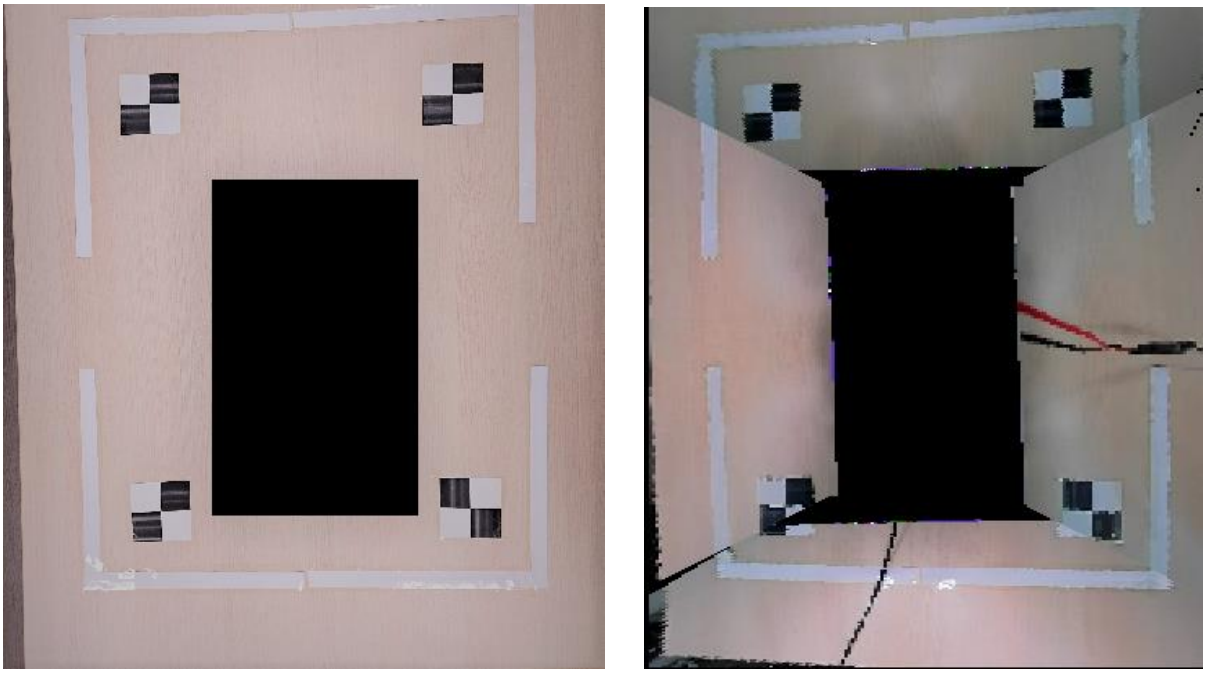
4.1. Kreiranje testnih video okvira za procjenu kvalitete dobivenih pogleda odozgo

Kako bi omogućili pravilno testiranje rješenja i prikaz ostvarenih rezultata implementiranih algoritama za generiranje pogleda odozgo, potrebno je izraditi smislenu testnu scenu okoline makete autića. Nakon što je scena izrađena, snima se okolina makete autića postavljene unutar izvedene scene. Okolina unutar koje automobil snima prikazana je na slici 4.1. te predstavlja referentni izgled okoline koju programsko rješenje treba što vjernije prikazati. Referentna slika je snimljena pomoću kamere mobilnog uređaja.



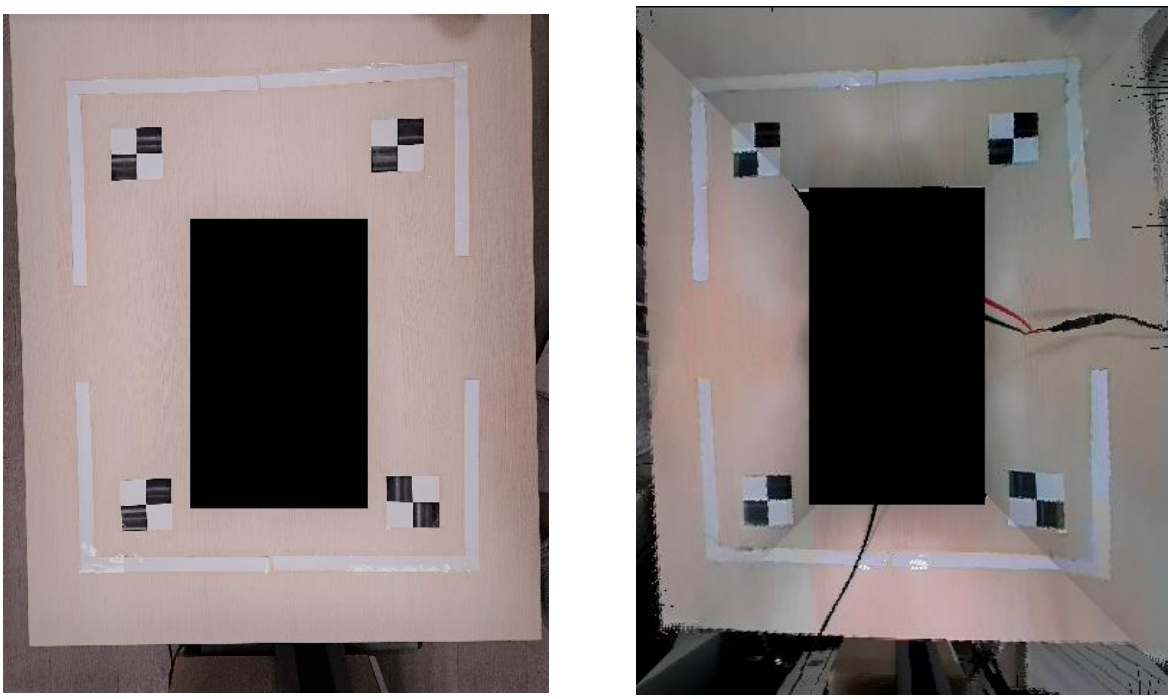
Slika 4.1. Referentni prikaz scene okruženja vozila snimljen mobilnim uređajem

Nakon što je izvedena scena, snimljeni su videozapisi pomoću razvojne ploče povezane s razvojnim računalom te internetskom vezom kako bi se videozapisi mogli spremiti i prikazati na računalu. Snimljeni su videozapisi okoline s prethodno izvedenim rješenjima s YUYV422I i YUV420SP formatima zapisa. Izdvojeni su pojedini okviri na kojima se vrši analiza i subjektivno ocjenjivanje kvalitete izrađenih prikaza. Za svaki izdvojeni okvir izrađena je referentna slika za istu prikazanu scenu koja služi za usporedbu i procjenu kvalitete dobivenog pogleda odozgo. Ovim postupkom se omogućava uvid u učinkovitost i funkcionalnost sustava. U nastavku su slikama 4.2. i 4.3. prikazani primjeri pojedinog referentnog i snimljenog okvira okoline za oba izvedena rješenja. Vidljivo je da su referentne slike različite veličine jer je u obzir uzeto skaliranje prikaza s YUV422I formatom zapisa. Zbog manje rezolucije došlo je do većeg gubitka elemenata slike pri rubovima te rješenje prikazuje manji dio okoline vozila u odnosu na prikaz s YUV420SP formatom zapisa.



a) b)

Slika 4.2. Prikaz okoline vozila (a) referentna slika snimljena mobilnim uređajem (b) prikaz s razvojne ploče s YUYV422I formatom zapisa



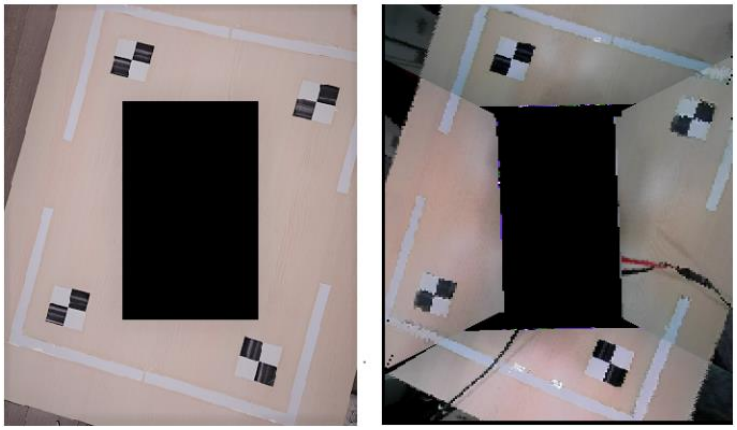
a) b)

Slika 4.3. Prikaz okoline vozila (a) referentna slika snimljena mobilnim uređajem (b) prikaz s razvojne ploče s YUV420SP formatom zapisa

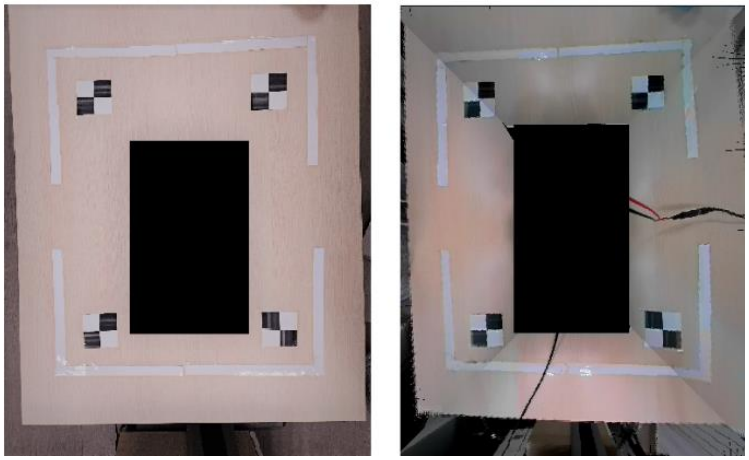
Za analizu i evaluaciju ostvarenih pogleda odozgo na vozilo izdvojeno je 20 okvira s odgovarajućim referentnim slikama. Osvrt na ove okvire pružit će uvid u prednosti, nedostatke i moguća poboljšanja izrađenih rješenja. Skup svih okvira korištenih u testiranju rada predloženih rješenja nalazi se u elektroničkom prilogu P.4.1. ovoga rada danom na DVD-u priloženom uz rad.

4.2. Subjektivna ocjena kvalitete slike jedinstvenog pogleda odozgo

Kako bi se provelo testiranje rada te subjektivno ocjenjivanje rezultata dobivenih rješenja, iskorišteno je 20 parova referentnih i dobivenih okvira spomenutih iznad. U nastavku su slikom 4.4. prikazana 3 para referentnih i dobivenih okvira. Slika 4.4. a) prikazuje par referentnog okvira i okvira dobivenog s razvojne ploče s YUYV422I formatom zapisa, dok slike 4.4 b) i c) prikazuje parove referentnih okvira i okvira dobivenih s razvojne ploče s YUV420SP formatom zapisa.



a)



b)

c)

Slika 4.4. Prikaz parova referentnih i dobivenih okvira (a) par s dobivenim okvirom s YUYV422I formatom zapisa (b) par s dobivenim okvirom s YUV420SP formatom zapisa (c) par s dobivenim okvirom s YUV420SP formatom zapisa

Za dobivanje subjektivne ocjene kvalitete pogleda odozgo potrebno je provesti anketu s određenim brojem gledatelja. U ovom slučaju provedena je anketa s deset ispitanika. Svaki ispitanik daje subjektivnu ocjenu zasnovanu na usporedbi dobivenih prikaza algoritma s referentnom slikom pogleda odozgo. Ispitanik daje po dvije ocjene za svaku sliku pogleda odozgo: ocjenu za cjelokupnu kvalitetu prikaza te ocjenu za kvalitetu spojenih rubova zasnovanu na poklapanju linija na mjestima gdje se prikazi sa susjednih kamera preklapaju. U nastavku je prikazana tablica 4.1. sa srednjim ocjenama pojedinačnih ispitanika za svih 20 parova okvira za rješenja koja rade s YUYV422I i YUV420SP formatima zapisa.

Tablica 4.1. Prosječne subjektivne ocjene ispitanika za prikaze rješenja

Ispitanici	Subjektivna ocjena za YUYV422I		Subjektivna ocjena za YUV420SP	
	Kvaliteta prikaza	Kvaliteta rubova	Kvaliteta prikaza	Kvaliteta rubova
Ispitanik 1	7,80	7,60	8,40	8,60
Ispitanik 2	6,60	6,80	8,60	8,60
Ispitanik 3	6,20	6,80	9,20	8,60
Ispitanik 4	7,00	8,00	8,80	8,80
Ispitanik 5	8,20	7,60	9,60	8,40
Ispitanik 6	6,60	6,00	8,40	7,60
Ispitanik 7	7,60	7,40	7,40	7,80
Ispitanik 8	7,60	7,00	8,00	7,80
Ispitanik 9	7,00	6,40	8,00	7,80
Ispitanik 10	6,40	6,40	7,40	6,80
Srednja ocjena	7,10	7,00	8,38	8,08
Ukupna srednja ocjena	7,05		8.23	

Razlika u prosječnim subjektivnim ocjenama ostvarenih prikaza između dvaju korištenih formata zapisa je uzrokovana skaliranjem prikaza s YUYV422I formatom zapisa u svrhu poboljšanja brzine izvođenja rješenja. Kako bi se i iz drugog ugla analizirale subjektivne ocjene dobivenih prikaza, potrebno je analizirati svaki par referentnih i dobivenih okvira kako bi se dobio uvid u razlike dobivenih ocjena. Tablicom 4.2. su prikazane prosječne subjektivne ocjene svih gledatelja za kvalitetu i spojene rubove svih 20 parova okvira za rješenje s YUYV422I formatom zapisa uz standardnu devijaciju ocjena koja služi kao mjera raspršenosti ocjena, dok tablica 4.3. prikazuje prosječne ocjene za parove okvira rješenja s YUV420SP formatom zapisa uz standardnu devijaciju.

Tablica 4.3. Prosječne subjektivne ocjene parova okvira s YUYV422I formatom zapisa

Parovi okvira	Srednja ocjena za kvalitetu prikaza	Standardna devijacija ocjena za kvalitetu prikaza	Srednja ocjena za kvalitetu rubova	Standardna devijacija ocjena za kvalitetu rubova
Par 1	6,90	0,20	6,90	0,10
Par 2	7,60	0,50	6,60	0,40
Par 3	7,20	0,10	6,90	0,10
Par 4	7,10	0,00	8,00	1,00
Par 5	6,70	0,40	6,60	0,40
Par 6	7,10	0,00	8,00	1,00
Par 7	6,90	0,20	6,90	0,10
Par 8	7,60	0,50	6,60	0,40
Par 9	7,20	0,10	6,90	0,10
Par 10	6,70	0,40	8,00	1,00

Tablica 4.2. Prosječne subjektivne ocjene parova okvira s YUV420SP formatom zapisa

Parovi okvira	Srednja ocjena za kvalitetu prikaza	Standardna devijacija ocjena za kvalitetu prikaza	Srednja ocjena za kvalitetu rubova	Standardna devijacija ocjena za kvalitetu rubova
Par 1	7,90	0,48	7,90	0,18
Par 2	8,20	0,18	6,70	1,38
Par 3	8,00	0,38	8,30	0,22
Par 4	8,30	0,08	8,10	0,02
Par 5	9,50	1,12	9,40	1,32
Par 6	8,50	0,12	7.90	0,18
Par 7	8,10	0,28	6,70	1,37
Par 8	7,90	0,48	8,30	0,22
Par 9	8,20	0,18	8,10	0,02
Par 10	7,90	0,48	9,40	1,32

Iz analize rezultata ankete proizlazi da su ostvarene prosječne subjektivne ocjene prikaza bolje uz korištenje YUV420SP formata zapisa u usporedbi s YUYV422I formatom. Korištenjem YUV420SP formata, koji uključuje manje skaliranje rezolucije, postignut je kvalitetniji prikaz s manje vidljivih nedostataka. Prvo, korištenje YUV420SP formata omogućilo je postizanje veće rezolucije (1280 x 720 elemenata slike) u odnosu na YUYV422I format koji ima smanjenu rezoluciju (640 x 360 elemenata slike). To znači da YUV420SP

format ima veću kvalitetu prikaza okoline vozila, što je ključno za sigurno parkiranje, jer vozač ima jasniji pregled okoline. Također, YUV420SP format dao je bolje rezultate kod spojenih rubova, gdje su prijelazi između različitih dijelova slike glađi i prikladno se poklapaju. Nasuprot tome, YUYV422I format je rezultirao većim gubitkom elemenata slike prilikom ispravljanja izobličenja i transformacije perspektive. To je dovelo do lošije kvalitete prikaza, gubitaka detalja pri rubovima i pojave „efekta bloka“ koji najviše narušavaju ocjenu ispitanika i jasni prikaz objekata u okolini vozila.

Odstupanje prikaza s desne strane automobila je imalo ključan utjecaj na subjektivnu ocjenu obaju algoritama, što je rezultiralo smanjenjem ukupne ocjene. Odstupanje je uzrokovano time što nije implementirana matrica homografije za desnu i stražnju kameru zbog manjka dostupne memorije razvojne ploče, tako da desna kamera mora biti postavljena pod točnim kutom i u ravnini s lijevom kamerom kako bi se osiguralo precizno poravnanje prikaza. Iz tog razloga, potrebno je obratiti posebnu pažnju na postavljanje kamera kako bi se osigurala što veća točnost i usklađenost prikaza.

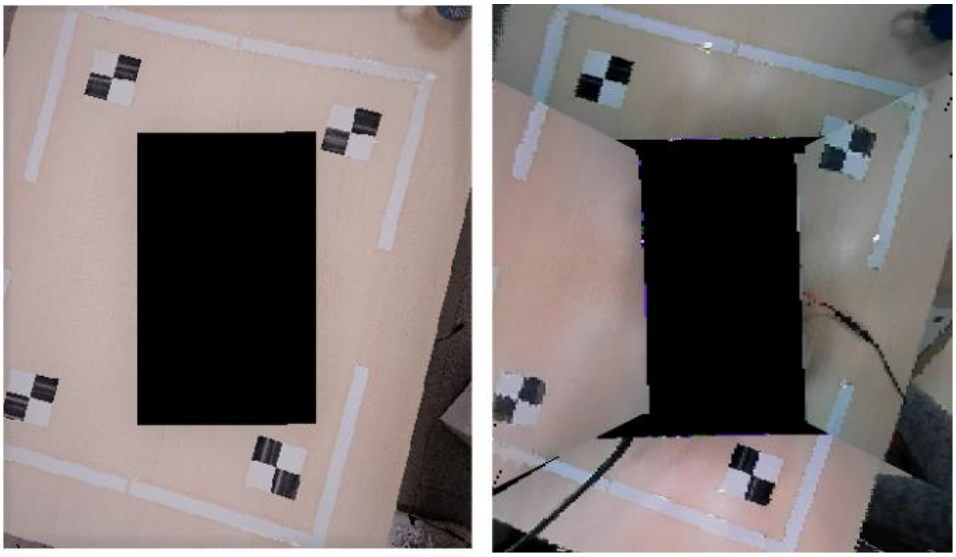
Fotometrijsko poravnanje nije uspješno implementirano na razvojnu ploču zbog ograničenosti dostupne memorije razvojne ploče, što je rezultiralo manjom ocjenom spojenih rubova kod obaju algoritama. Dolazi do različitih svjetlosnih uvjeta i nijansi boja između pojedinih prikaza te je jasno vidljiva granična linija između susjednih prikaza. Takvi spojevi su vizualno uočljivi te utječu na ukupnu kvalitetu izvedenog prikaza.

Detaljnijom analizom tablica 4.2. i 4.3. koje sadrže prosječne ocjene dobivenih okvira moguće je bolje razumjeti rezultate provedenog testiranja i obrazložiti prethodno navedene prednosti i nedostatke pojedinih rješenja te vezu između kvalitete prikaza i rubova sa standardnom devijacijom ocjena. Standardna devijacija pojedinih parova okvira je veća kod rješenja s YUV420SP formatom zapisa, što ukazuje na veću varijabilnost ocjena među različitim sudionicima ankete te na prisutnost različitih preferencija ili osjetljivosti na određene aspekte prikaza među sudionicima, koja je izraženija na rješenju s kvalitetnijim prikazom.

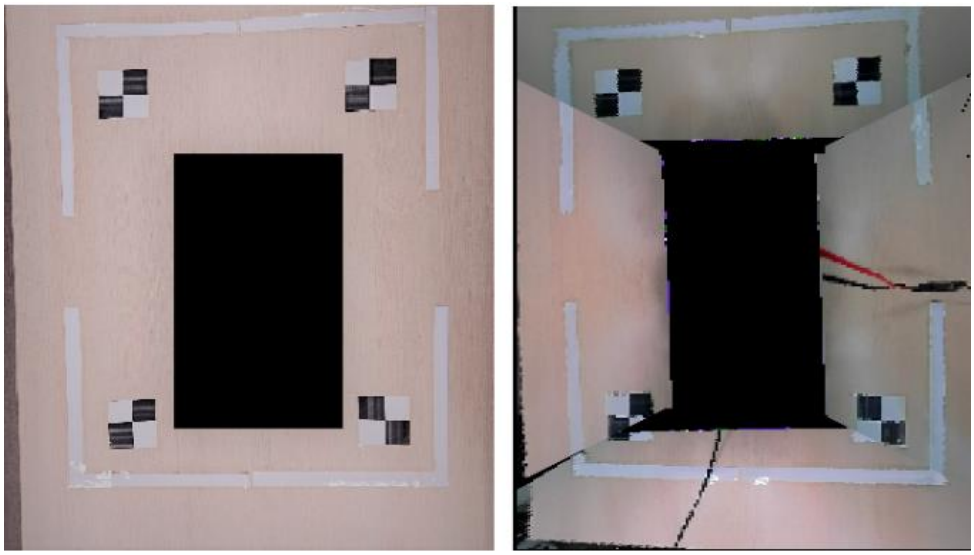
Prikazi s YUYV422I formatom zapisa dobili su niže prosječne ocjene. Ova opažanja su usklađena s prethodno iznesenim zaključcima o lošijoj kvaliteti prikaza i rubova za ovaj format. Unatoč lošijim rezultatima, primjetna je manja standardna devijacija u usporedbi s YUV420SP formatom. Ovo ukazuje na relativno veću konzistentnost u percepciji kvalitete među

sudionicima. S obzirom na sudionike koji su izloženi niskim standardima kvalitete, pojava je takva da oni percipiraju odstupanja prikaza manje intenzivno. Drugim riječima, ako slika nije visoke kvalitete, sudionici ne primjećuju manje razlike i odstupanja koja su „prikrivena“ zbog niske kvalitete prikaza i manjka detalja slike. U nastavku je slikom 4.5. prikazan par okvira s najvećom srednjom ocjenom kvalitete prikaza i rubova, slikom 4.6. par okvira s najvećim negativnim odstupanjem subjektivne ocjene od prosječne ocjene kvalitete prikaza dok je slikom 4.7. prikazan par okvira s najvećim negativnim odstupanjem od prosječne ocjene kvalitete rubova.

Slika 4.5. Prikaz para referentnog i dobivenog okvira s YUYV422I formatom zapisa s najvećom prosječnom subjektivnom ocjenom



Slika 4.6. Prikaz para referentnog i dobivenog okvira s YUYV422I formatom zapisa s najvećom standardnom devijacijom subjektivne ocjene kvalitete prikaza



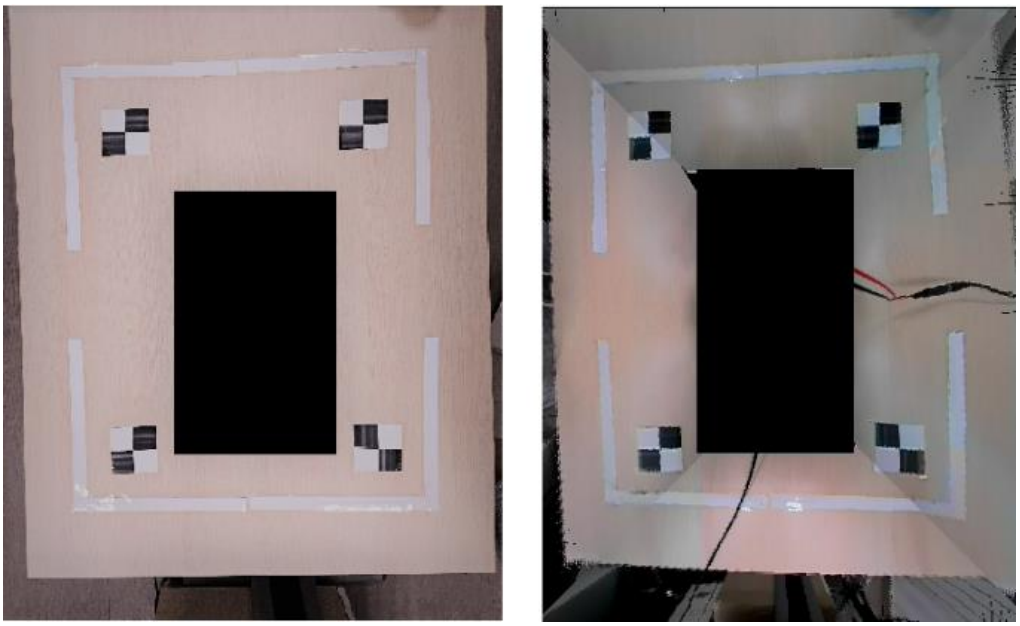
Slika 4.7. Prikaz para referentnog i dobivenog okvira s YUYV422I formatom zapisa s najvećom standardnom devijacijom subjektivne ocjene kvalitete rubova

Iz prikazanih slika vidljive su razlike dobivenih okvira nastale zbog nesavršenosti mjerenja i različitih svjetlosnih uvjeta okoline. Slika 4.5. predstavlja najbolje ocjenjenu dobivenu sliku iz razloga što su susjedni prikazi dobro poravnati te je manja potreba za fotometrijskim poravnanjem zbog svjetlosti okoline koja je ujednačenija nego na ostalim snimljenim slikama. Prikaz je dobre kvalitete te su rubovi dobro usklađeni, s manjim odstupanjima. Slika 4.6. predstavlja dobiveni prikaz s najmanjom ocjenom kvalitete prikaza u usporedbi s referentnom slikom. Vidljiva je jasna potreba za fotometrijskim poravnanjem zbog različitih svjetlosnih uvjeta različitih strana automobila te je vidljivo odstupanje nijansi boja susjednih prikaza. Dolazi do jasnije vidljive pojave „efekta bloka“ jer su bijele linije scene pri rubovima, gdje je najveći gubitak elemenata slike. Slika 4.7. prikazuje dobiveni prikaz s najmanjom ocjenom kvalitete spojenih rubova. S obzirom da standardna devijacija nije velika te dobivena ocjena nije puno manja od srednje ocjene, vidljivo je manje odstupanje na rubovima, pogotovo desne kamere za koju nije implementirana matrica homografije te samim time nije uspješno poravnata. Na svim slikama je vidljiva potreba za fotometrijskim poravnanjem, što bi zasigurno povećalo kvalitetu prikaza. Zaključno, analizom prikazanih slika očite su varijacije u dobivenim okvirima koje proizlaze iz nesavršenosti izvođenja snimanja okoline, raznolikih svjetlosnih

uvjeta okoline i potrebe za implementacijom fotometrijskog poravnanja i matrice homografije za sve četiri kamere.

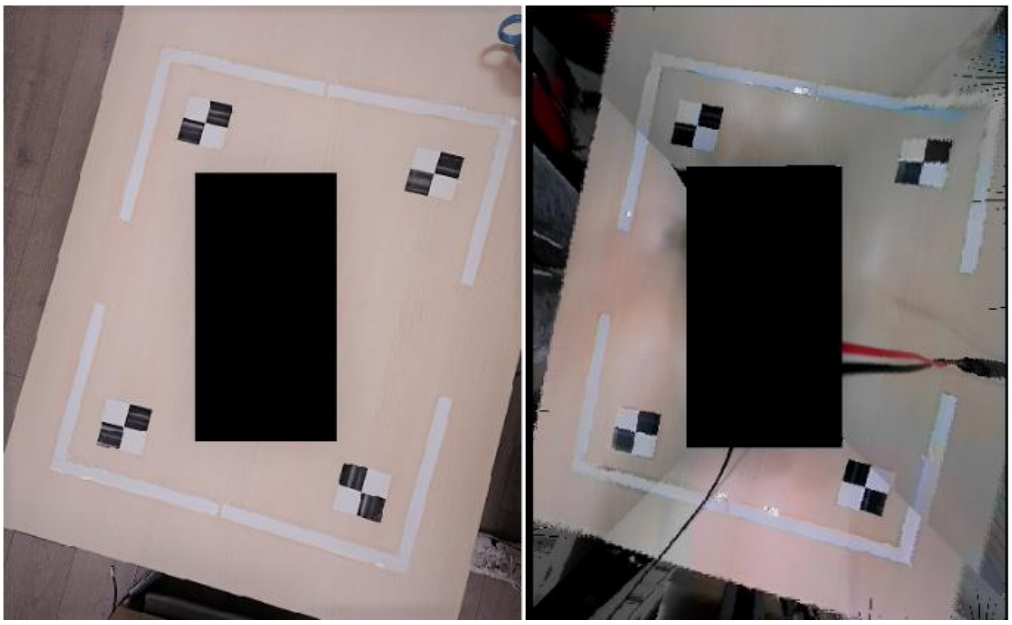
Prikazi s YUV420SP formatom zapisa dobili su bolje prosječne ocjene. To je u skladu s prethodno opisanim opažanjima da je ovaj format omogućio veću rezoluciju i bolje rezultate kod spojenih rubova. Prosječna standardna devijacija subjektivnih ocjena veća je u odnosu na YUYV422I format, što može ukazivati na manju konzistentnost u percepciji kvalitete među sudionicima. Sudionici izloženi visokim standardima kvalitete mogu intenzivnije percipirati i više uočavati manja odstupanja od visoke kvalitete prikaza, što dovodi do manje subjektivne ocjene u odnosu na prosječnu ocjenu. To znači da je njihova sklonost primjećivanju sitnih detalja i kritičnost prema manjim promjenama veća u odnosu na situaciju kad je manja kvaliteta prikaza.

U nastavku je slikom 4.8. prikazan par okvira s najvećom srednjom ocjenom kvalitete prikaza i rubova, slikom 4.9. par okvira s najvećim negativnim odstupanjem subjektivne ocjene od prosječne ocjene kvalitete prikaza dok je slikom 4.10. prikazan par okvira s najvećim negativnim odstupanjem od prosječne ocjene kvalitete rubova.



Slika 4.8. Prikaz para referentnog i dobivenog okvira s YUV420SP formatom zapisa s najvećom prosječnom subjektivnom ocjenom

Slika 4.10. Prikaz para referentnog i dobivenog okvira s YUV420SP formatom zapisa s najvećom standardnom devijacijom subjektivne ocjene kvalitete prikaza



Slika 4.9. Prikaz para referentnog i dobivenog okvira s YUV420SP formatom zapisa s najvećom standardnom devijacijom subjektivne ocjene kvalitete rubova

Iz prikazanih slika vidljive su razlike dobivenih okvira. Na najbolje ocjenjenoj slici, prisutna svjetlosna situacija okoline omogućava jasan prikaz okoline vozila bez izražene potrebe za fotometrijskim poravnanjem koje nije implementirano. Rubovi su dobro spojeni s minimalnim odstupanjima, što doprinosi ukupnom dojmu prikaza. Slika 4.9. koja je dobila

najmanju ocjenu za kvalitetu, prikazuje jasnu potrebu za fotometrijskim poravnanjem zbog prisutnosti različitih razina svjetlosti oko automobila. Na slici su uočljivi šumovi nastali prilikom interpolacije te dolazi do odstupanja boja susjednih prikaza. Ovakvi faktori znatno utječu na percepciju i rezultiraju nižom ocjenom. Slika 4.10. najbolje prikazuje izazove vezane za matrice homografije i fotometrijsko poravnanje. Nedostatak implementirane matrice homografije za desnu i zadnju kameru je očit. Došlo je do znatnog odstupanja u poravnanju slika, pogotovo s desne strane automobila do kojeg je došlo zbog nesavršenosti izvođenja snimanja okoline i manjeg pomaka kamere s desne strane automobila. Prisutna je potreba za fotometrijskim poravnanjem zbog svjetlosti okoline. Ocjene i devijacije u kvaliteti između ovih slika jasno reflektiraju utjecaj navedenih faktora, naglašavajući važnost svakog pojedinog elementa u stvaranju kvalitetnog vizualnog dojma.

Analizom subjektivnih ocjena rješenja može se zaključiti da su gledatelji obično kritičniji prema ocjenama kvalitete rubova. To je zbog činjenice da su rubovi bitan aspekt u doživljaju detalja i cjelokupne ravnoteže slike. Loši rubovi često mogu biti vizualno primjetni i mogu utjecati na opću percepciju prikaza. Ocjena kvalitete prikaza obuhvaća sve aspekte kvalitete slike, uključujući svjetlosne uvjete, rezoluciju, boje, detalje i rubove. Odnos između ocjena kvalitete prikaza i ocjena kvalitete rubova može se prikazati Pearsonovim koeficijentom korelacije, koji predstavlja statističku mjeru za određivanje linearnog odnosa dviju varijablu. Za tablicu 4.2., Pearsonov koeficijent korelacije iznosi -0.35. To znači da postoji slab negativan linearni odnos između prosječnih ocjena kvalitete prikaza i ocjena kvalitete rubova svih parova okvira. Za tablicu 4.3., Pearsonov koeficijent korelacije iznosi 0.34. To ukazuje na slab pozitivan linearni odnos između ocjena kvalitete prikaza i ocjena kvalitete rubova. Zaključuje se da se ocjene kvalitete samih rubova ne mogu pouzdano koristiti kao zamjena za ukupnu ocjenu kvalitete prikaza. Iako ocjene rubova pružaju uvid u doživljenu kvalitetu spajanja rubova, one ne mogu adekvatno zamijeniti ukupnu ocjenu kvalitete prikaza koja uključuje i druge karakteristike slike poput boje, kontrasta, oštrine i slično. To je potkrijepljeno niskim apsolutnim iznosom Pearsonovog koeficijenta korelacije koji ukazuje na odsutnost značajne linearnosti između ovih dviju mjera.

Zaključno, uzimajući u obzir sve navedene činjenice, jasno je da korištenje YUV420SP formata pruža bolje rezultate prikaza sustava u odnosu na YUYV422I format što se tiče kvalitete dobivenog prikaza. Rezultati ankete evaluacije rješenja priloženi su u P.4.2. i P.4.3.

4.3. Analiza vremena izvođenja predloženog rješenja

Kako bi se dodatno analizirale kvaliteta slike pogleda odozgo i brzina izvođenja razvijenih algoritama, potrebno je izmjeriti brzinu rada istih. Kao što je ranije spomenuto, omogućeno je praćenje statistika izvođenja *UseCase*-a pritiskom znaka „*p*“ prilikom izvođenja algoritma u *Tera Term* okruženju. Prati se količina dobivenih okvira prilikom snimanja te se mjeri broj „izgubljenih“ okvira tokom izvođenja pojedinog dijela *UseCase*-a i mjeri se vrijeme potrebno za pojedinu funkcionalnost. U analizi dobivenih rješenja potrebno je pročitati vrijednost broja ulaznih FPS. Kao što je ranije spomenute, kamere snimaju 30 FPS. Nakon što okviri dođu do algoritma koji ih obrađuje, dolazi do gubitka okvira jer je vrijeme pristizanja novih okvira kraće od vremena obrade i transformacije okvira.

Prilikom obrade prikaza s formatom YUYV422I dolazi do gubitka oko 23 FPS jer je potrebno oko 150ms za izvršavanje svih funkcionalnosti algoritma. Dalje se ti okviri procesiraju te na izlaz prikaza na zaslonu dolazi 6.5 FPS. Statistike tokom praćenja ulaznih vrijednosti te izlaznih vrijednosti prikazane su na slikama 4.11. i 4.12.

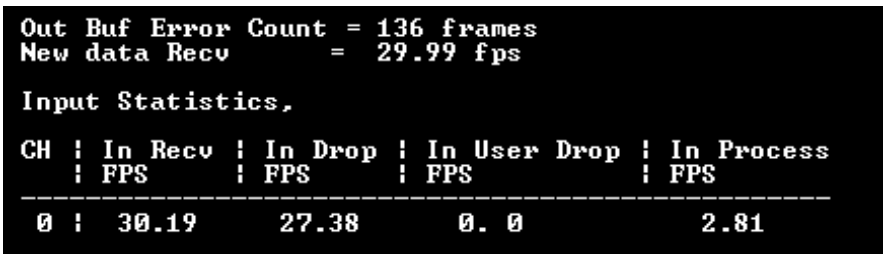
```
Out Buf Error Count = 105 frames
New data Recv       =  29.97 fps

Input Statistics,

CH  | In Recv  | In Drop  | In User Drop | In Process
    | FPS      | FPS      | FPS          | FPS
-------------------------------------------------------
 0  |  30.19      23.31       0. 0           6.88
```

Slika 4.11. Statistika ulaznih okvira YUYV422I formata

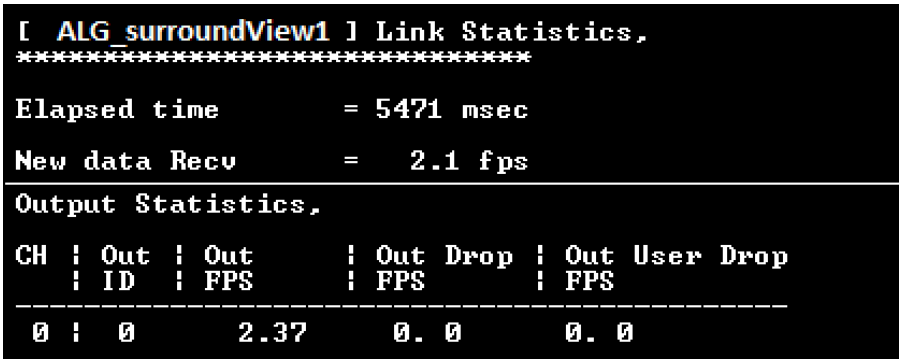
```
### CPU [  DSP1], LinkID [ 44],

[ ALG_surroundView ] Link Statistics,
*********************************

Elapsed time        = 5009 msec

New data Recv       =   6.18 fps

Output Statistics,

CH  | Out  | Out      | Out Drop  | Out User Drop
    | ID   | FPS      | FPS       | FPS
-----------------------------------------------
 0  |  0       6.58      0. 0       0. 0
```

Slika 4.12. Statistika izlaznih okvira YUYV422I formata

Prilikom obrade prikaza s formatom YUV420SP dolazi do gubitka oko 27 FPS jer je potrebno oko 350 ms za izvršavanje svih funkcionalnosti algoritma. Dalje se ti okviri procesiraju te na izlaz prikaza na zaslonu dolazi 2.5 FPS. Treba naglasiti kako je prikaz u rješenju s formatom YUV420SP dvostruko veće rezolucije te je iz tog razloga zauzeće memorije te zahtjevnost obrade samog algoritma veće. Statistike tokom praćenja ulaznih vrijednosti te izlaznih vrijednosti prikazane su na slikama 4.13. i 4.14.



Slika 4.14. Statistika ulaznih okvira YUV420SP formata



Slika 4.13. Statistika izlaznih okvira YUV420SP formata

Razlika u potrebnim resursima i vremenu izvođenja je očita. Algoritam za ostvarivanje pogleda odozgo predstavlja poprilično zahtjevan sustav te prilikom prolaska kroz pojedine dijelove zahtjeva određenu procesorsku moć. Najzahtjevniji dio izvedenog rješenje je promjena perspektive prikaza s pojedinih kamera, gdje se za svaku kameru vrši transformacija svih elemenata slike za implementirane homografije te zahtjeva dodatnu interpolaciju elemenata slike koji se „gube“ prilikom izvođenja procesa. Izmjereno vrijeme izvođenja prebacivanja perspektive jednog prikaza s kamere iznosi oko 20 ms, što je previše za ostvarivanje vremenski efikasnog sustava u stvarnom vremenu.

S obzirom da je rješenje usmjereno na pomoć pri parkiranju, pristup s 3 do 7 FPS može biti prihvatljiv jer u tom scenariju vozač ne vozi brzo i ne zahtijeva visok broj FPS za kvalitetan vizualni prikaz. Povećanjem broja FPS može se postići još vremenski glađi i fluidniji prikaz okoline, što bi vozaču omogućilo još veću sigurnost i praktičnost pri parkiranju. Dakle, izvedena rješenja mogu biti prikladna i zadovoljavajuća za svoju svrhu pomoći pri parkiranju.

4.4. Osvrt na predložena rješenja i dobivene rezultate

Predložena vlastita rješenja za dobivanje pogleda odozgo na vozilo zasnovana su na korištenju dvaju formata zapisa: YUV420SP i YUYV422I. Analiza subjektivnih ocjena i rezultata ankete ukazuje na to da YUV420SP format pruža bolju kvalitetu prikaza u odnosu na YUYV422I format. Razlika u ocjenama proizlazi iz izvedene kvalitete prikaza i skaliranja istih kako bi se poboljšale performanse izvedbe.

Kada se koristi YUV420SP format zapisa, algoritam ne koristi skaliranje rezolucije te pruža kvalitetniji prikaz. S druge strane, rješenje zasnovano na YUYV422I formatu zapisa pokazuje lošiju kvalitetu prikaza s pojavom „efekta bloka“. Budući da su Y, U i V komponente smještene u blokovima, pri ispravljanju i zakretanju pojedinih prikaza dolazi do gubitka elemenata slike, a rubovi i prijelazi s više detalja nisu dovoljno glatki.

Treba naglasiti da testiranje algoritma za dobivanje pogleda odozgo na maketi autića može otkriti neke nesavršenosti i izazove do kojih dolazi prilikom implementacije takvih sustava. Iako se algoritam može dobro izvoditi na razvojnoj ploči, postoje neki faktori koji mogu utjecati na preciznost i točnost dobivenih rezultata.

Jedan od glavnih izazova je mali pomak kamere u odnosu na ravninu koju snima. Pomak može imati značajan utjecaj na rezultate. Čak i manje promjene u poziciji kamere mogu dovesti do promjene perspektive i geometrijske transformacije slike. To može utjecati na točnost generiranog pogleda odozgo, jer se mali pomak može prenijeti na konačnu sliku, pogotovo na maketi autića gdje su kamere na maloj udaljenosti od ravnine površine okoline koju prikazuju.

Drugi faktor koji može utjecati na testiranje je kut snimanja. U slučaju *fish-eye* kamera s širokim vidnim poljem, promjena kuta snimanja može rezultirati različitim izobličenjima i perspektivama na slici. To može utjecati na ispravljanje izobličenja i transformaciju perspektive, što može dovesti do nepreciznosti u generiranju konačnog pogleda odozgo.

Razlika u performansama između korištenja dvaju formata ima utjecaj na ukupnu učinkovitost sustava za prikaz okoline vozila. Ključni faktor koji uzrokuje ovu razliku je razlučivost slike. Korištenje YUYV422I formata s rezolucijom 640 x 360 elemenata slike omogućuje postizanje brže brzine obrade (6.5 FPS) u odnosu na YUV420SP format s rezolucijom 1280 x 720 elemenata slike koji postiže brzinu obrade od 2.5 FPS.

Za izvorni ulazni video od 30 FPS, gubitak od 23.5 FPS za YUYV422I format i 27.5 FPS za YUV420SP format može imati značajan utjecaj na stvarno vrijeme prikaza okoline vozila. Ovi gubici proizlaze iz vremena potrebnog za izvršavanje svih funkcionalnosti algoritma te gubitak ovog broja FPS može dovesti do vidljivog zastoja i niže glatkoće prikaza na zaslonu. To može otežati vozaču u situacijama kada se zahtijeva brza reakcija i jasan pregled okoline.

Kako bi se poboljšala ukupna učinkovitost sustava i smanjio gubitak okvira, potrebno je razmotriti optimizaciju i unapređenje algoritama te mogućnost nadogradnje sustava ili implementacije na naprednije sustave s više dostupnih resursa. Može se razmotriti optimizacija koda i korištenje naprednih tehnika obrade slika poput onih koje pruža *OpenCV* biblioteka koja se dodaje kao programska podrška, kako bi se postigla bolja efikasnost. Također, razmatranje drugih formata zapisa i rezolucija koji bolje odgovaraju zahtjevima sustava može biti korisno. U konačnici, rješenje s YUV420SP formatom zapisa može biti preferirano za ovu primjenu kako bi se osigurao stabilniji prikaz okoline vozila, pružajući vozaču jasniji pregled i veću sigurnost pri parkiranju.

Preložena rješenja su pokazala određene prednosti i nedostatke u pogledu kvalitete prikaza i brzine obrade. Daljnja optimizacija algoritama, nadogradnja razvojne ploče i programske podrške ili implementacija na naprednije sustave mogu rezultirati poboljšanjem kvalitete prikaza i performansi, čime se otvara mogućnost bolje integracije u autonomne sustave i poboljšanje ukupnog iskustva korisnika.

5. ZAKLJUČAK

U ovom diplomskom radu opisani su algoritmi kalibracije kamere i generiranja pogleda odozgo prostora oko automobila, koji su uspješno implementirani na AMV Alpha ADAS razvojnu ploču. Rješenje pomaže vozačima pri parkiranju automobila ili pri sporoj vožnji u neposrednoj blizini objekata i prepreka koje je teško vidjeti iz vozila. Generirani prikaz prostora oko automobila zasniva se na korištenju četiriju širokokutnih kamera s vidnim poljem od 180°.

Proces kalibracije kamere predstavlja dobivanje unutarnjih i vanjskih parametara kamere potrebnih za ispravljanje izobličenja prikaza nastalih *fish-eye* lećom. Pomoću unutarnjih parametara kamere ispravljaju se izobličenja slike i prikazuje se stvarna slika okoline. Nakon što su prikazi ispravljeni, slijedi proces generiranja pogleda prostora oko automobila. Vrši se transformacija perspektive okvira sa svake kamere u ptičju perspektivu. U ovom koraku se također obavlja rotacija potrebnih okvira kako bi se pravilno postavili na konačnoj slici. Konačna slika se formira spajanjem i pozicioniranjem slika sa svake kamere na odgovarajućim lokacijama, ovisno o njihovom vidnom polju.

Važno je naglasiti da se u algoritmu neki podaci fiksiraju kako bi se ubrzao proces generiranja pogleda prostora oko automobila. To uključuje fiksiranje ispravljanja izobličenja slike, transformacije perspektive i položaja slike sa svake kamere na konačnoj slici. Ovo je moguće zbog činjenice da se kamere nalaze na točnom određenom položaju na automobilu, što omogućava predefinirane parametre.

Implementacija algoritma na AMV Alpha ADAS razvojnoj ploči pokazala je zadovoljavajuće rezultate, ali je važno napomenuti da ploča ima ograničenu memoriju i slabiju programsku podršku, što je rezultiralo određenim nedostacima. Algoritam uspješno ispravlja izobličenja slike dobivene širokokutnom kamerom, što ukazuje na to da su parametri kalibracije kamere točni. Također, generiranje pogleda prostora oko automobila daje dobre rezultate za definirane parametre transformacije i određenu poziciju kamere na vozilu. Manjak memorije razvojne ploče na kojoj su izvedeni algoritmi onemogućuje razvoj fotometrijskog poravnanja te kvalitetnije i brže izvođenje programskog rješenje. Također, rezultati dobivenih prikaza odstupaju od referentnih prikaza zbog nesavršenosti izvođenja testiranja i vanjskih utjecaja kao na primjer mali pomak kamere prilikom snimanja okoline.

Iz subjektivnih rezultata ankete vidljivo je da korištenje YUV420SP formata zapisa pruža bolje prosječne subjektivne ocjene prikaza u usporedbi s YUYV422I formatom. Gledatelji su posebno uočili da YUV420SP format omogućuje višu rezoluciju (1280 x 720 elemenata slike), što povećava pregled okoline vozila i sigurnost parkiranja. Također, YUV420SP format daje bolje rezultate kod spojenih rubova s glatkim prijelazima, dok YUYV422I format pokazuje nedostatke poput gubitka detalja i „efekta bloka“. Odstupanje prikaza na desnoj strani automobila negativno je utjecalo na ocjene za oba algoritma, što ukazuje na potrebu za preciznim poravnanjem kamera. Neuspješno implementirano fotometrijsko poravnanje rezultiralo je vidljivim graničnim crtama između susjednih prikaza, što utječe na ukupnu kvalitetu prikaza.

Korištenje ovog sustava moguće je u vozilima opremljenim AMV Alpha ADAS pločom. Važno je napomenuti da testiranje na maketi autića može pružiti korisne informacije i smjernice, ali konačna primjena algoritma u stvarnom automobilu može se razlikovati. Algoritam zahtjeva prilagodbu različitim tipovima vozila zbog korištenja različitih kamera te pozicija postavljenih kamera na vozilu. Stoga je važno provesti testiranje u različitim uvjetima.

Konačno, implementirani algoritmi za generiranje pogleda odozgo pružaju vozačima korisne informacije o okolini vozila. Unatoč manjim nedostacima uzrokovanim ograničenom memorijom i neuspješnim fotometrijskim poravnanjem, algoritam je pokazao svoju vrijednost i potencijal za daljnje unapređenje u stvarnim uvjetima na vozilima opremljenim ovom pločom. Za daljnji napredak i poboljšanje algoritma, potencijalno je potrebno unaprijediti samu opremu (engl. *hardware)* ploče i programsku podršku na njoj kako bi se osigurala veća dostupna memorija i bolja izvedba.

LITERATURA

[1] Understanding Lens Distortion, https://learnopencv.com/understanding-lens-distortion/, pristup ostvaren 03.07.2023.

[2] Camera Calibration, https://docs.opencv.org/4.5.2/dc/dbb/tutorial_py_calibration.html, pristup ostvaren 03.07.2023.

[3] Hartley, R., & Zisserman, A. (2004). Multiple View Geometry in Computer Vision (2nd ed.). Cambridge: Cambridge University Press. doi:10.1017/CBO9780511811685

[4] Developing a real-time social distancing detection system based on YOLOv4-tiny and bird-eye view for COVID-19 - Scientific Figure on ResearchGate. Dostupno na: https://www.researchgate.net/figure/Perspective-transformation-to-bird-eye-view-a-perspective-view-imageb-Perspective_fig5_358796118 pristup ostvaren 01.07.2023.

[5] Y. Liu and B. Zhang, „Photometric alignment for surround view camera system,“ 2014 IEEE International Conference on Image Processing (ICIP), Paris, France, 2014, pp. 1827-1831, doi: 10.1109/ICIP.2014.7025366.

[6] X. Yan, L. Wang and Y. Li, „A Calibration Method for the Surround-View Parking Assistant System,“ 2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, 2020, pp. 485-488, doi: 10.1109/ICSESS49938.2020.9237687.

[7] D. Buljeta, M. Vranješ, Z. Marčeta and J. Kovačević, „Surround view algorithm for parking assist system,“ 2019 Zooming Innovation in Consumer Technologies Conference (ZINC), Novi Sad, Serbia, 2019, pp. 21-26, doi: 10.1109/ZINC.2019.8769410.

[8] K. Lee, D. Kim, H. Do, J. Kim and K. Chae, „Effective Photometric Alignment for Surround View Monitoring System,“ 2019 IEEE 9th International Conference on Consumer Electronics (ICCE-Berlin), Berlin, Germany, 2019, pp. 390-391, doi: 10.1109/ICCE-Berlin47944.2019.8966185.

[9] B. Kragulj, S. Laza, M. Milosevic, and G. Velikic, “One solution of complex ADAS HW system testing,” in 2017 25th Telecommunication Forum (TELFOR), Belgrade, Nov. 2017, pp. 1–4. doi: 10.1109/TELFOR.2017.8249477.

[10] Diagram of YU12, I420, YV12, NV12, NV21, YUV420P, YUV420SP, YUV422P, YUV444P, https://www.programmerall.com/article/41702038544/, pristup ostvaren 05.07.2023.

[11] OpenCV: Open Source Computer Vision Library, https://opencv.org/, pristup ostvaren 03.07.2023.

[12] NumPy: Fundamental package for scientific computing with Python, https://numpy.org/, pristup ostvaren 03.07.2023.

[13] Imutils: A series of convenience functions to make basic image processing functions such as translation, rotation, resizing, and displaying images with Matplotlib easier, https://github.com/jrosebr1/imutils, pristup ostvaren 03.07.2023.

[14] PIL (Python Imaging Library), https://pillow.readthedocs.io/en/stable/, pristup ostvaren 03.07.2023.

[15] Dataset – Multi-Camera Pose Estimation, https://mscvprojects.ri.cmu.edu/2022team12/dataset-2/, pristup ostvaren 17.07.2023.

[16] Dissecting the Camera Matrix, Part 2: The Extrinsic Matrix, https://ksimek.github.io/2012/08/22/extrinsic/, pristup ostvaren 17.07.2023.

[17] W. Li, F. Qu, Y. Wang and P. Liu, „A fast vehicle top-view system,“ The 2014 2nd International Conference on Systems and Informatics (ICSAI 2014), Shanghai, China, 2014, pp. 389-393, doi: 10.1109/ICSAI.2014.7009320.R. Szeliski, Computer Vision. London: Springer London, 2011. doi: 10.1007/978-1-84882- 935-0.

[18] Correcting Fisheye Images, https://www.baeldung.com/cs/correcting-fisheye-images, pristup ostvaren 05.07.2023.

[19] „RDACM24B Automotive Camera Platform Technical Datasheet“, IMI D&D, 2015.

[20] Camera View, https://www.neodynamic.com/Products/Help/ImageDrawSDKNET2.0/action_cameraview.htm, pristup ostvaren 15.08.2023.

[21] A flexible vehicle surround view camera system by central-around coordinate mapping model, dostupno na: https://www.researchgate.net/figure/In-a-surround-view-camera-system-four-fish-eye-cameras-are-mounted-around-the-vehicle_fig1_328164472, pristup ostvaren 17.08.2023.

SAŽETAK

Diplomski rad opisuje algoritme za dobivanje pogleda odozgo na vozilo koji se mogu koristiti u sustavu za autonomna vozila. U radu su opisane važne smjernice i koraci za uspješnu kalibraciju kamere. Algoritam za generiranje pogleda prostora oko automobila definira geometrijske operacije nad slikom kako bi se generirao cjeloviti i povezani pogled odozgo. Ove operacije uključuju ispravljanje izobličenja, transformaciju perspektive i spajanje slika. Algoritam se uspješno izvodi kada su kamere postavljene na vozilo s poznatim kalibracijskim parametrima i kada se nalaze na predviđenoj poziciji na vozilu. Važno je osigurati točnost kalibracijskih parametara i pravilno postavljanje kamera kako bi se postigao pouzdan i precizan pogled prostora oko vozila. Algoritam izrađen s YUV420SP formatom zapisa s brzinom obrade od 2.5 FPS za rezoluciju 1280 x 720 elemenata slike dobio je bolje subjektivne ocjene ispitanika od YUYV422I formata s brzinom obrade od 6.5 FPS za rezoluciju 640 x 360 elemenata slike. Daljnji napredak ovog algoritma može se postići nadogradnjom samog hardvera ploče i programske podrške kako bi se osigurala veća dostupna memorija i poboljšala ukupna izvedba sustava.

Ključne riječi: autonomna vozila, izobličenje slike, kalibracija kamere, pogled odozgo, povezivanje slika, širokokutna kamera, transformacija perspektive

AN ALGORITHM FOR OBTAINING A TOP VIEW OF A VEHICLE

ABSTRACT

The thesis describes algorithms for obtaining a top-down view of a vehicle that can be used in an autonomous vehicle system. The work outlines important guidelines and steps for successful camera calibration. The algorithm for generating a surround view of the vehicle defines geometric operations on the image to create a comprehensive and connected top-down view. These operations include distortion correction, perspective transformation, and image stitching. The algorithm performs successfully when cameras are mounted on the vehicle with known calibration parameters and positioned as intended. Ensuring the accuracy of calibration parameters and proper camera placement is crucial to achieving a reliable and precise surround view. The algorithm implemented with the YUV420SP format, processing at 2.5 frames per second for a resolution of 1280 x 720, received better subjective ratings from the participants compared to the YUYV422I format, processing at 6.5 frames per second for a resolution of 640 x 360. Further advancement of this algorithm can be achieved by improving the hardware of the board and the software support to provide increased available memory and enhance the overall system performance.

Keywords: autonomous vehicles, image distortion, camera calibration, top-down view, image stitching, wide-angle camera, perspective transformation

ŽIVOTOPIS

Ivan Marinić rođen je 03. kolovoza 1999. godine u Slavonskom Brodu. Pohađao je Osnovnu školu „Bogoslav Šulek“ u Slavonskom Brodu. Nakon završene osnovne škole, upisao je Tehničku školu Slavonski Brod. Nakon završetka srednje škole, 2018. godine upisuje preddiplomski sveučilišni studij Elektrotehnike na FERIT-u, gdje se opredjeljuje za izborni blok „Komunikacije i informatika“. U rujnu 2021. godine završava preddiplomski studij i upisuje diplomski studij komunikacija i informatike na istom fakultetu.

PRILOZI

P.4.1. – Mapa s videozapisima i okvirima ostvarenih rezultata

P.4.2. – Mapa sa skupom slika korištenih za provedbu evaluacijske ankete

P.4.3. – Mapa s rezultatima evaluacijske ankete programskog rješenja