

SVEUČILIŠTE U ZAGREBU  
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1122

**Detekcija zatvaranja petlje u  
SLAM-u temeljenom na  
podvodnim vizualnim podacima**

Aleksandar Bibulić

Zagreb, srpanj 2015.

*Umjesto ove stranice umetnite izvornik Vašeg rada.*  
*Da bi ste uklonili ovu stranicu obrišite naredbu \izvornik.*



# SADRŽAJ

<b>1. Uvod</b>	<b>1</b>
<b>2. Detekcija zatvaranja petlje</b>	<b>2</b>
2.1. Hashiranje video okvira . . . . .	3
2.1.1. Opisna matrica projekcije . . . . .	3
2.1.2. Ortogonalna projekcija . . . . .	5
2.2. Potvrda detekcije zatvorenosti petlje temeljene na hash funkciji . . . .	5
2.2.1. RANSAC ( <i>engl. Random sample consensus</i> ) algoritam . . . .	6
2.2.2. Pretraživanje sličnosti slika . . . . .	8
2.2.3. Provjera zatvaranja petlje . . . . .	9
2.2.4. SIFT algoritam za izdvajanje značajki (deskriptora) . . . . .	10
2.2.5. SURF ( <i>engl. speeded up robust features</i> ) . . . . .	15
<b>3. Lokalizacija kamere u prostoru</b>	<b>16</b>
3.1. Model kamere . . . . .	16
3.1.1. Kalibracija stereo kamere . . . . .	20
3.1.2. Epipolarna geometrija . . . . .	21
3.1.3. Esencijalna i fundamentalna matrica . . . . .	22
3.1.4. Rektifikacija stereo para slika . . . . .	23
3.1.5. Triangulacija i matrica dispariteta . . . . .	25
3.2. Vizualna odometrija . . . . .	27
3.2.1. Pronalaženje korespondentnih značajki . . . . .	27
3.2.2. Estimacija gibanja . . . . .	29
<b>4. Eksperimentalni rezultati</b>	<b>31</b>
4.1. Rezultati zatvaranja petlje . . . . .	31
4.2. Rezultati lokalizacije kamere u prostoru . . . . .	32
<b>5. Zaključak</b>	<b>38</b>





# 1. Uvod

Napretkom tehnologije sve je intenzivnije istraživanje okruženja u kojem se nalazimo. Od kopnenih površina, mora i morskih dubina pa sve do svemirskih prostranstva. Istraživanje svakog pojedinog područja predstavlja karakteristične izazove. Iako vodene površine čine 70 posto ukupne Zemljine površine, tek je mali dio podmorja istražen. Razlozi tome su brojni, od velikih hidrostatskih tlakova na većim dubinama do otežane komunikacije ispod vode.

Za razliku od istraživanja svemira gdje je komunikacija na velike udaljenosti omogućena odašiljanjem elektromagnetskih signala, istraživanje podmorja je otežano upravo iz razloga jer voda zbog svojih fizikalnih svojstava iznimno jako prigušuje elektromagnetske signale te otežava komunikaciju.

Zbog sve većeg interesa za korištenje autonomnih podvodnih vozila (*engl. Autonomous Underwater Vehicle*), javlja se potreba za lokalizaciju istih ispod vode. Kako je konvencijalna metoda komunikacije nemoguća ispod vode, onemogućeni su i GPS (*engl. Global Positioning System*) sustavi. Javlja se potreba za alternativnim rješenjima lokalizacije autonomnih podvodnih ronilica. Jedno od rješenja je korištenje akustičkih uređaja za komunikaciju te ujedno i za lokalizaciju same ronilice. Takva rješenja lokalizacije su uvelike nepraktična i u velikom slučaju iznimno skupa.

Jeftinija alternativa jest SLAM odnosno simultana lokalizacija i mapiranje (*engl. Simultaneous localization and mapping*) koja koristi stereo kameru u procesu određivanja pozicije autonomne podvodne ronilice. Zbog mogućnosti pogreške u određivanju pozicije autonomne podvodne ronilice pomoću stereo kamere dodatno se koriste algoritmi koji omogućuju detekciju zatvaranja petlje te pružaju dodatnu sigurnost za precizniju lokalizaciju.

## 2. Detekcija zatvaranja petlje

Mogućnost detekcije zatvaranja petlje iznimno je bitna u podvodnim sustavima gdje je autonomnoj podvodnoj ronilici potrebno prepoznati okolinu u kojoj se nalazi te odrediti je li njeno okruženje u potpunosti novo ili se je ronilica prethodno već našla na istoj poziciji. Potreba za korištenjem takvog sustava se javlja u slučajevima gdje postoji vjerojatnost pogreške senzora koji se koriste za dinamičko pozicioniranje. Detekcija zatvorenosti petlje u tom slučaju pruža ispravljanje pozicije ronilice odnosno lokalizaciju. Algoritmi koji se koriste u detekciji zatvaranja petlje često zahtijevaju veliku računalnu moć i veliki memorijski prostor te se javlja potreba za novim efikasnijim rješenjima koji bi smanjili vrijeme izvršavanja algoritma, smanjili računalnu zahtjevnost i povećali preciznost samog prepoznavanja zatvaranja petlje.

U ovom radu korišten je algoritam koji koristi hash funkciju na video okvirima u stvarnom vremenu. Hash funkcija svaki video okvir konvertira u karakterističan opis korištenjem originalnih slikovnih značajki. Reducirana memorijska zahtjevnost opisa slikovnih značajki omogućuje ovom algoritmu veliku efikasnost u aspektu brzine izvođenja samog algoritma. Glavne odlike algoritma su:

1. Minimiziranje vremena pretraživanja video okvira koji su potencijalni kandidati za zatvaranje petlje.
2. Algoritam ne zahtijeva fazu treniranja koja bi mogla negativno utjecati na rezultate u izvršavanju procesa lokalizacije.
3. Izvršavanje algoritma u stvarnom vremenu.
4. Reducirana potrebna memorija za kvalitetno izvršavanje algoritma zahvaljujući mogućnosti spremanja opisa značajki svakog pojedinog video okvira.

## 2.1. Hashiranje video okvira

Osnovni princip hash funkcije je da veliku skupinu podataka sažme na manju, odnosno da iz velike skupine podataka izdvoji string fiksne duljine. Skupovi podataka mogu biti oblika tekstualne poruke, dokumenta ili u ovom slučaju slikovna datoteka. Hash funkciju možemo opisati na sljedeći način:

$$h = \Psi(\rho) \quad (2.1)$$

gdje je  $\Psi$  hash funkcija,  $\rho$  je skup ulaznih podataka a  $h$  je izračunati izlazni string na skupu ulaznih podataka.

Slikovne hash funkcije se najčešće dijele u dvije skupine:

1. Hash string se računa iz tekstu slike, informacije o bojama, transformacija u frekvencijskom području ili iz faktorizacijske matrice [22] [13].
2. Hash funkcija zasnovana na značajkama slike odnosno ključnih dijelova (deskriptora) video okvira [17] [19] [21].

Hash funkcija korištena u radu bazirana je na značajkama slike te svaki video okvir ima jedinstveni set značajki koje ga karakteriziraju. Razlog zašto je korišten takav pristup leži u tome da je hash izračunat iz značajki slika invarijantan na rotaciju, translaciju i skaliranje, odnosno moguće je prepoznati zatvaranje petlje snimanjem iste scene s različitih kuteva i udaljenosti. Značajke se izdvajaju odabirom SIFT ili SURF algoritma koji će biti objašnjeni kasnije u poglavlju.

### 2.1.1. Opisna matrica projekcije

Opisnu matricu definiramo kao  $D_{n \times m}$ , gdje je  $n$  broj vizualnih značajki određenih u trenutnom video okviru, a  $m$  je broj odgovarajućih deskriptora za te značajke. Broj  $m$  se postavlja na fiksnu vrijednost tijekom eksperimenta, dok broj  $n$  varira po iznosu za svaki video okvir. Cilj postupka je da se transformira veličinu deskriptora u fiksnu veličinu:  $\mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{k \times m}$ . Veličina broja  $k$  mora biti prihvatljiva s obzirom na memorijsku zahtjevnost i visoku performansu.

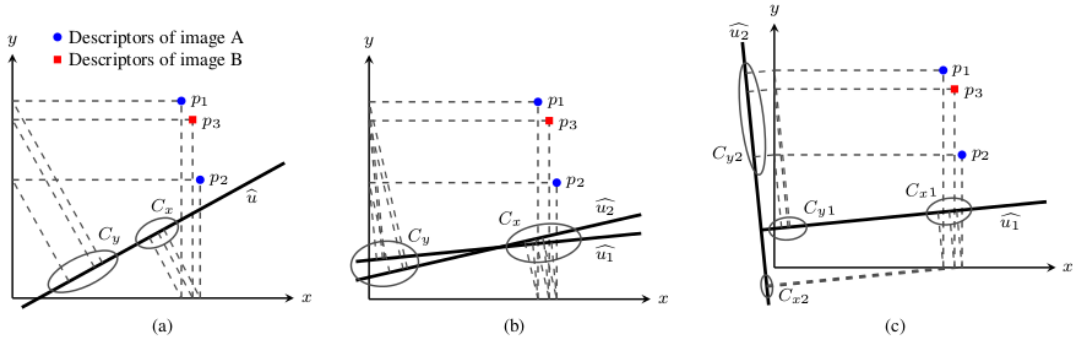
Projekcija opisne matrice na jedinični vektor  $\hat{u}$  proizvoljnog smjera uz ( $k = 1$ ) se definira na sljedeći način:

$$\rho_i = \sum_{j=1}^n D(j, i) \cdot \hat{u}(j) \quad (2.2)$$

gdje je  $1 \leq i \leq m$ . Izlazni string je u tom slučaju jednak:

$$h = \rho_1 + \rho_2 + \dots + \rho_{m-1} + \rho_m. \quad (2.3)$$

Smjer jediničnog vektora  $\hat{u}$  se definira na samom početku izvršavanja procesa i ostaje isti za proračun hash stringova svih video okvira dobivenih za vrijeme izvršavanja podvodne misije autonomne podvodne ronilice. Duljina jediničnog vektora  $\hat{u}$  mora biti prilagodljiva broju redaka  $n$  opisne matrice. Maksimalna vrijednost broja  $n$  je u samom početku nepoznata. Iz tog razloga se duljina jediničnog vektora  $\hat{u}$  postavlja na što veći broj, ali se pri tome mora paziti da se ne uspori izvršavanje algoritma. Prihvatljiva vrijednost duljine jediničnog vektora  $\hat{u}$  je oko 5% veličine slike u broju piksela.



**Slika 2.1:** Idealan slučaj (a) gdje se sličnosti i razlike deskriptora preslikaju na točke projekcije, dva proizvoljno usmjerena vektora (b), ortogonalni vektori (c)

Na slici 2.1(a) možemo vidjeti projekciju deskriptora dvije različite slike A i B, na jedinični vektor proizvoljnog smjera. Pretpostavimo da deskriptor ima  $x$  i  $y$  komponentu. Projekcije  $x$  komponente na jedinični vektor  $\hat{u}$  označene su sa skupom  $C_x$  i nisu od koristi u razlikovanju deskriptora  $p_2$  od  $p_3$ . Razlog tome je što su deskriptori  $p_2$  i  $p_3$  nedovoljno udaljeni po  $x$  komponentu. Međutim, projekcija  $y$  komponente na jedinični vektor  $\hat{u}$  jasno označava razliku između deskriptora  $p_2$  i deskriptora  $p_3$ , odnosno postoji veća udaljenost između projekcija. Skup projekcija  $y$  komponente na jedinični vektor  $\hat{u}$  označavamo sa  $C_y$ . Na slici 2.1(b) vidimo mogućnost da ni jedan od mogućih proizvoljnih jediničnih vektora  $\hat{u}_1$  i  $\hat{u}_2$  ne daje dobre rezultate u razlikovanju deskriptora. Iz tog razloga najčešće se sljedeći jedinični vektor u nizu odabire tako da bude ortogonalan s prethodnim.

### 2.1.2. Ortogonalna projekcija

U svrhu preciznijeg određivanja zatvaranja petlje, opisna matrica se projicira u više smjerova i te se projekcije ulančavaju kako bi tvorile završni hash string. Sa  $h_l$  možemo označiti rezultat projekcije opisne matrice na  $l$ -ti smjer jediničnog vektora  $\hat{u}_l$ . Jednadžbu 2.3 tada možemo zapisati u obliku:

$$h_l = \sum_{i=1}^m \sum_{j=1}^n D(j, i) \cdot \hat{u}_l(j) \quad (2.4)$$

te je finalni hash string definiran kao:

$$H = \sum_{l=1}^k h_l \quad (2.5)$$

Skup vektora  $\hat{u}_l (1 \leq l \leq k)$  definiramo prije početka izvršavanja algoritma za prepoznavanje zatvaranja petlje. Ukoliko bi se svi jedinični vektori generirali proizvoljno postojala bi vjerojatnost da su neki od vektora usmjereni u istom smjeru što bi dovelo da viška nepotrebnih informacija te bi se smanjila performansa algoritma. Taj problem se rješava tako da se odredi smjer jediničnog vektora, te je svaki sljedeći jedinični vektor ortogonalan na njemu prethodni. Primjer takvog rasporeda možemo vidjeti na slici 2.1(c), gdje su jedinični vektori  $\hat{u}_1$  i  $\hat{u}_2$  međusobno ortogonalni. Vidimo da projekcije  $y$  komponente na jedinični vektor  $\hat{u}_1$  ne možemo dobro razlikovati ali je zato projekcije na jedinični vektor  $\hat{u}_2$  vrlo jednostavno razlikovati. Za hash funkciju sastavljenu od  $k$  projekcija, prvi jedinični vektor  $\hat{u}_1$  se proizvoljno određuje, a svaki sljedeći mora biti ortogonalan na prethodne jedinične vektore. Ortogonalnost se određuje tako da njihov produkt bude jednak nuli:

$$\hat{u}_1(1) \cdot \hat{u}_2(1) + \hat{u}_1(2) \cdot \hat{u}_2(2) + \dots + \hat{u}_1(n) \cdot \hat{u}_2(n) = 0 \quad (2.6)$$

S obzirom na to da znamo orijentaciju jediničnog vektora  $\hat{u}_1$ , jediničnom vektoru  $\hat{u}_2$  možemo dodijeliti proizvoljne vrijednosti za indekse  $\hat{u}_2(1), \hat{u}_2(2), \dots, \hat{u}_2(n-1)$ , te zadnji indeks postaviti tako da vrijedi  $\hat{u}_1 \perp \hat{u}_2$ . Odabir  $k = 3$  pruža dobar odnos između veličine hash stringa i preciznosti detektiranja zatvaranja petlje.

## 2.2. Potvrda detekcije zatvorenosti petlje temeljene na hash funkciji

Proces hashiranja video okvira opisan u prethodnom potpoglavlju koristi se kako bi pronašao i izdvojio najbolje kandidate za zatvaranje petlje iz dosad snimljenih video

okvira. Nakon izdvajanja kandidata, provodi se validacija iz razloga prihvatanja ili odbacivanja odabranih kandidata. Algoritam se može podijeliti u dvije skupine:

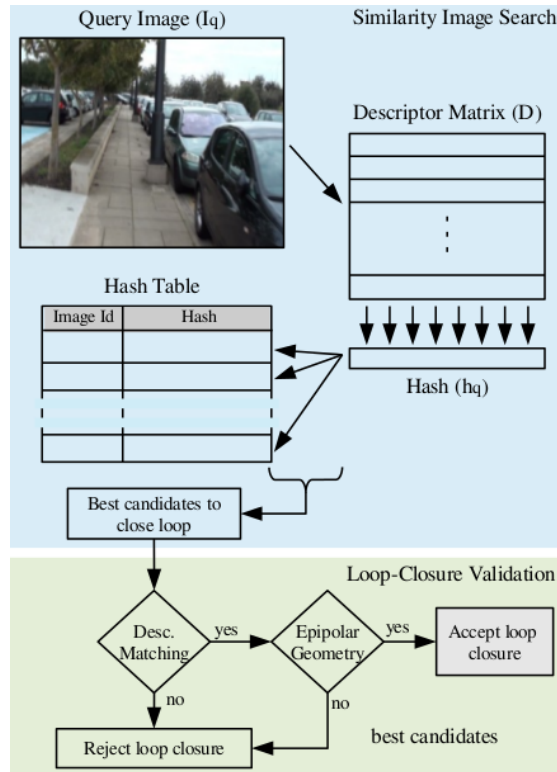
1. **Pretraživanje sličnosti slika:** za svaku novu sliku (video okvir) snimljenu tijekom misije autonomne podvodne ronilice, računa se hash string i sprema u hash tablicu. Zadnje dobivena slika se uspoređuje sa svim prethodnim snimkama osim sa zadanim brojem susjednih slika i sama sa sobom. Usporedba se vrši unutar hash tablice u svrhu pronalaska najboljih kandidata. Najbolji kandidati su one slike koje imaju sličan hash string. Nakon pronalaska kandidata, kako bi se rezultati dodatno poboljšali, koristi se vjerojatnosna funkcija koja koristi prošle informacije trajektorije autonomne podvodne ronilice.
2. **Potvrda zatvaranja petlje:** u ovom dijelu se svaki prethodno određeni kandidat uspoređuje sa trenutnim video okvirom, odnosno skup deskriptora se uspoređuje s deskriptorima trenutnog video okvira korištenjem Knn-Match (*engl. K-nearest neighbor Match*) algoritma. Nadalje koristimo epipolarno ispravljanje (epipolarna geometrija) između kandidata i trenutnog video okvira te se određeni kandidati izbacuju korištenjem RANSAC algoritma. Epipolarno ispravljanje će biti objašnjeno u daljnjem tekstu.

Cjelokupni postupak izvršavanja algoritma za prepoznavanje zatvaranja petlje je prikazan na slici 2.2.

### 2.2.1. RANSAC (*engl. Random sample consensus*) algoritam

RANSAC ili konsenzus slučajnih uzoraka (*engl. Random sample consensus*) je iterativna metoda koja se koristi za estimaciju matematičkih modela iz skupa podataka koji sadrže *outliere*, odnosno opzervacijske točke koje su udaljene od ostalih promatranih podataka [10] [8]. Pomak se javlja iz razloga varijabilnosti u postupku mjerenja ili može ukazivati na pogreške u samom mjerenju. RANSAC algoritam prepoznaje *outliere* u skupu podataka i estimira željeni model koristeći podatke koji ne sadrže *outliere*. Koraci provedbe RANSAC algoritma su sljedeći:

1. Proizvoljno odabiranje podskupa  $s$  iz skupa promatranih podataka  $S$  na temelju kojih se izgrađuje model.
2. Određivanje seta podataka  $S_i$  koji se nalaze unutar graničnih vrijednosti  $t$  modela.  $S_i$  je privremeni set ispravnih podataka iz seta  $S$ .



**Slika 2.2:** Prikaz procesa određivanja zatvaranja petlje

3. Ako je broj ispravnih podataka u setu  $S_i$  veći od granične vrijednosti  $T$ , pokreće se proces reestimacije modela koristeći samo podatke iz seta  $S_i$  i algoritam se završava.
4. Ako je broj ispravnih podataka u setu  $S_i$  manji od granične vrijednosti  $T$  vrati se na točku broj 1 u algoritmu.
5. Nakon  $N$  iteracija odabire se najveći set  $S_i$  ispravnih podataka po kojima se reestimira model.

Broj iteracija  $N$  se odabire tako da se s vjerojatnošću  $p$  osigura da je barem jedan od  $s$  slučajnih uzoraka točno određen. Čest odabir je  $p = 0.99$ . Označimo li s  $w$  vjerojatnost da je bilo koji podatak točno određen, tada je  $\epsilon = 1 - w$  vjerojatnost da je taj podatak pogrešno određen. Uz  $N$  odabira podataka dobivamo relaciju  $(1 - w)^N = 1 - p$  i vrijedi sljedeće:

$$N = \frac{\log(1 - p)}{\log(1 - (1 - \epsilon)^s)} \quad (2.7)$$

Oblik adaptivnog RANSAC algoritma prikazan je u sljedećim koracima: Promatramo  $N > \text{broj pokušaja}$ .



1. Odabiremo set podataka  $S$  iz kojeg na temelju modela izdvajamo set ispravnih podataka  $S_i$ .
2. Izračunava se vrijednost  $\epsilon = 1 - \frac{\text{broj ispravnih podataka}}{\text{ukupan broj podataka}}$ .
3. Određuje se  $N$  prema jednadžbi 2.7.
4. Povećava se broj pokušaja za 1.

### 2.2.2. Pretraživanje sličnosti slika

Svaki novi hash string ( $h_q$ ) prikupljen tijekom trajektorije misije se uspoređuje s prethodno spremljenim hashovima unutar hash tablice. Svrha usporedbe je traženje potencijalnih kandidata za zatvaranje petlje. Hashovi koji su u okolini  $s$ -najbližih susjeda trenutno dobivenog video okvira se ne razmatraju kao kandidati za zatvaranje petlje iz razloga da se izbjegne pogrešno detektiranje između uzastopnih snimaka iste scene. Hash  $h_q$  se uspoređuje samo sa hashovima  $h_1, h_2 \dots h_{q-s}$ . Parametar  $s$  se podešava ovisno o frekvenciji snimanja video okvira (*engl. frame rate*), te brzini kretanja kamere.

Usporedba hashova se izvršava primjenom  $l_1$ -norme između trenutnog video okvira i kandidata za zatvorenost petlje na sljedeći način:  $\|h_q - h_c\|_1$ , gdje je  $h_c$  hash kandidat za zatvaranje petlje. Odabiru se kandidati s najmanjom vrijednosti  $l_1$  i prosljeđuju u sljedeći korak validacije. Za poboljšavanje kvalitete rada algoritma koriste se informacije prijedene trajektorije. Ako pomoću  $l_1$ -norme odredimo za trenutni hash  $h_q$  najboljeg kandidata za zatvorenost petlje  $h_i$ , susjedi hasha  $h_i$  imaju veliku vjerojatnost da također budu kandidati za zatvaranje petlje. Ovakvu vjerojatnost susjednih kandidata možemo prikazati normalnom razdiobom. Označimo li sa  $S_q$  slučajnu varijablu koja predstavlja pretpostavku za zatvaranje petlje trenutnog video okvira  $I_q$ , tada izraz  $S_q = i$  označava događaj da je video okvir  $I_q$  prepoznat kao zatvaranje petlje s video okvirom  $I_i$ , dok izraz  $S_q = -1$  označava događaj da nije prepoznato zatvaranje petlje. Označimo li s  $I_c$  potencijalni video okvir za zatvaranje petlje nakon računanja  $l_1$ -norme, a s  $I^s$  skup  $s$  susjednih video okvira u okolini  $I_c$  ( $I^s = \{I_{c-s}, \dots, I_{c+s}\}$ ), možemo zapisati Gaussovu razdiobu vjerojatnosti da je  $I_q$  prepoznat kao zatvaranje petlje sa  $I_c$  na sljedeći način:

$$p(S_q | I_b^s) = \eta \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(I_b^s - I_c)^2}{2\sigma^2}} \quad (2.8)$$

gdje je  $I_b^s$   $b$ -ti element skupa  $I^s$ ,  $\eta$  je normalizacijski faktor,  $\sigma$  je amplituda normalne razdiobe odnosno standardna devijacija i ovisi o preklapanjima između slika. Iz 2.8

vidimo da je svakom video okviru skupa  $I^s$  pridružena vjerojatnost za zatvaranje petlje s video okvirom  $I_q$ . Vjerojatnost da skup  $I^s$  zatvara petlju s  $I_q$  dan je izrazom:

$$p(S_q|I^s) = [p(S_q|I_{c-s}^s), \dots, p(S_q|I_{c+s}^s)] \quad (2.9)$$

Poželjno je imati više od jednog kandidata za zatvaranje petlje pa iz tog razloga definiramo skup  $Z_p$  kao set od  $p$  najboljih video okvira razmatranih nakon usporedbe  $l_1$ -normom, odnosno  $Z_p = \{I_{c0}, I_{c1}, \dots, I_{cp-1}\}$ . Slijedi da je vjerojatnost zatvaranja petlje između trenutnog video okvira  $I_q$  i video okvira u susjedstvu  $p$  najboljih kandidata (uračunavši i  $p$  najboljih kandidata) sljedeća:

$$p(S_q|Z_p^s) = \sum_{j=0}^{p-1} p(S_q|I^{s_j}) \quad (2.10)$$

gdje je s  $j$  označen indeks kandidata.

Finalno, sa  $Z_p^s(q)$  možemo označiti skup susjeda svih najboljih  $p$  kandidata za zatvaranje petlje s trenutnim video okvirom  $I_q$ . Moguće je uzeti u obzir informacije o zatvorenosti petlje video okvira snimljenih netom prije trenutnog video okvira kako bi izračunali vjerojatnost da trenutni video okvir  $I_q$  zatvara petlju s bilo kojim video okvirom  $I_i$  koristeći sljedeću jednadžbu:

$$p(S_q|I^I) = \sum_{\omega=0}^{\nu} p(S_q|Z_p^s(q - \omega)) \quad (2.11)$$

gdje je  $I^I = \{I_0, I_1, \dots, I_{q-s}\}$  i  $\eta$  je broj prijašnjih neposredno snimljenih video okvira koji se uzimaju u obzir za računanje konačne vjerojatnosti.

Konačni rezultat algoritma vraća vektor s indeksima video okvira  $p$  najboljih kandidata koji daju  $p$  najboljih rezultata prema formuli 2.11. Kandidati se u sljedećoj fazi prihvaćaju ili odbacuju. Vrijednost  $p$  mora biti pomno odabrana kako bi se uskladio odnos između efikasnosti algoritma i njegove preciznosti. Za velike iznose varijable  $p$  vrijeme izvođenja algoritma se povećava. Razlog toma je veći broj *deskriptora* koji se moraju provjeravati i potrebe za računanjem epipolarne geometrije  $p$  puta, ali je preciznost algoritma veća. Za male vrijednosti varijable  $p$  vrijeme provedbe algoritma se značajno reducira, ali na štetu lošijeg prepoznavanja zatvaranja petlje.

### 2.2.3. Provjera zatvaranja petlje

Za svaki od  $p$  najboljih kandidata odabranih u prethodnom poglavlju provjeravaju se *deskriptori* sa *deskriptorom* značajki trenutnog video okvira. Ako je broj podudarnih *deskriptora* veći od prethodno zadanog *thresholda*, *outlieri* se odbacuju po principu

epipolarne geometrije koristeći formulu  $xFx' = 0$ , gdje je  $F$  fundamentalna matrica,  $x$  je vektor koji sadržava značajke trenutnog video okvira,  $x'$  je vektor koji sadržava značajke kandidata. Nakon toga se provodi RANSAC algoritam koji je prethodno objašnjen. Dva video okvira čije značajke zadovolje epipolarna ograničenja, vrlo vjerojatno prikazuju istu scenu gledanu iz različitih gledišta. Ako je proces provjere dosljedan, tada se zatvorenost petlje prihvaća kao točna detekcija. Pretraživanja sličnosti slika ima linearnu složenost  $O(kmT)$ , gdje je  $T$  veličina *hash tablice*

Provjera zatvaranja petlje je zadatak koji oduzima veći dio računalnih resursa i ovisi o odabiru broja kandidata  $p$ , tipu *deskriptora* i veličini video okvira. Vrijeme izvršavanja provjere ne ovisi o duljini trajektorije autonomne podvodne ronilice, odnosno o ukupnom broju slika.

## 2.2.4. SIFT algoritam za izdvajanje značajki (deskriptora)

SIFT ili transformacija značajki neovisna o skaliranju (*engl. scale-invariant feature transform*) je algoritam koji detektira lokalne značajke unutar slike [15]. Glavne prednosti SIFT algoritma su to što je invarijantan na skaliranje, rotaciju, iluminaciju i translaciju. Algoritam se sastoji od sljedećih točaka:

1. **Detekcija ekstrema kroz niz skala** (*engl. Scale-space extreme detection*): Algoritam traži točke u video okvirima koje bi zadovoljile uvjete ponavljanja, stabilnosti i invarijantnosti. Invarijantnost točaka na različite skale dobiva se pretraživanjem slikovnih značajki na različitim skalama. Različite skale dobivamo primjenom Gaussovog filtra. Prostorna skala slike je definirana funkcijom  $L(x, y, \sigma)$  koja je produkt konvolucije Gaussovog filtra  $G(x, y, \sigma)$  i slike  $I(x, y)$ :

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \quad (2.12)$$

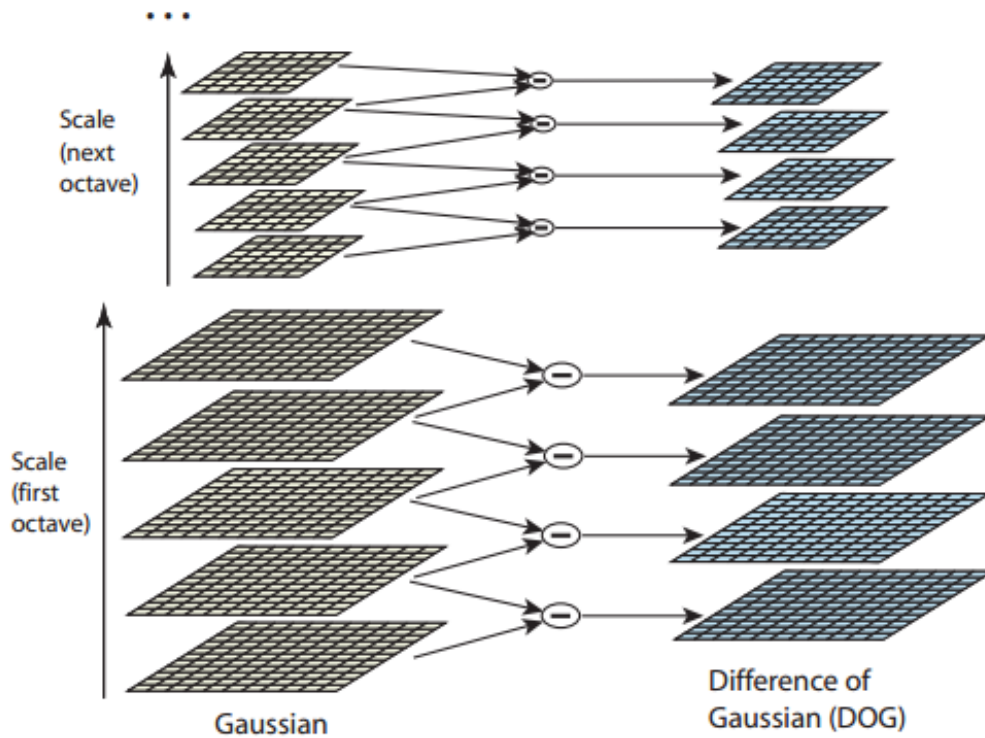
gdje  $*$  označava konvoluciju. Gaussov filtar je definiram izrazom:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}. \quad (2.13)$$

Kako se potencijalne značajke traže na rubovima slike moguće je koristiti Laplaceov rubni operator. Radi bržeg izvođenja algoritma umjesto Laplaceovog operatora rubova koristi se razlika zamućenih slika s Gaussovim filtrom (*engl. Difference of Gaussian; DoG*). DoG računamo kao konvoluciju trenutnog video okvira i razlike dviju susjednih slika zamućenih Gaussovim filtrom:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned} \quad (2.14)$$

Grafički prikaz DoG algoritma možemo vidjeti na slici 2.3.



**Slika 2.3:** Izgradnja DoG piramide. Za svaku oktavu kreira se grupa slika zamućenih Gaussovom filtrom (prikazano lijevo) i iz njih se oduzimanjem susjednih kreira DoG piramida (prikazano desno). Svakim prelaskom u novu oktavu nova ulazna slika postaje dva puta manja nego prethodna.

2. **Traženje ekstrema:** Kako bi prepoznali lokalni minimum i maksimum od  $D(x, y, \sigma)$ , svaki uzorak slike se uspoređuje s njegovim 8-susjedstvom u trenutnom video okviru i s 9-susjedstvom u skali iznad i ispod. Značajka se skalira ukoliko je veća ili manja od susjeda. Računska zahtjevnost ovog koraka nije velika iz razloga jer se većina značajki eliminira nakon nekoliko provjera.
3. **Lokalizacija značajki:** Nakon detekcije ekstrema potrebno je preračunati njihove cjelobrojne lokacije, translacije i skale u realne. To pridonosi većoj stabilnosti i preciznijoj lokalizaciji značajki. Proračun je temeljen na 3D kvadratnoj funkciji koju razvijemo u Taylorov red:

$$D(x) = D + \frac{\partial D^T}{\partial x} x + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x. \quad (2.15)$$

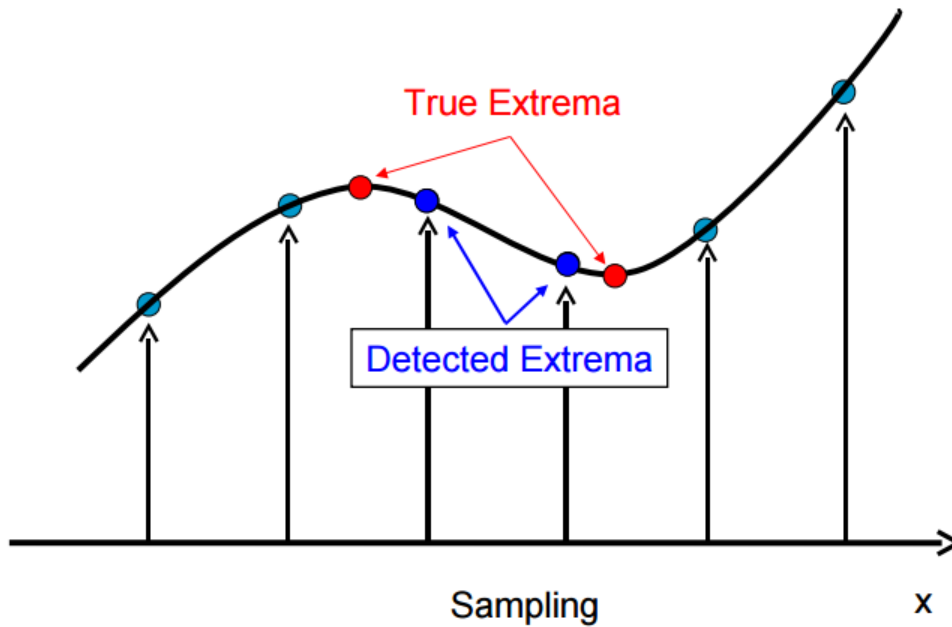
Interpolirane parametre ekstrema dobivamo derivacijom dane funkcije po para-

metrima ekstrema  $\mathbf{x} = (x, y, \sigma)^T$  i izjednačavanjem s nulom:

$$\hat{x} = -\frac{\partial D^{-1}}{\partial x^2} \frac{\partial D}{\partial x} \quad (2.16)$$

gdje je  $\frac{\partial D^{-1}}{\partial x^2}$  Hessianova matrica.

Ako je vektor  $\hat{x} = (\hat{x}, \hat{y}, \hat{\sigma})$  po nekoj od svojih vrijednosti veći od 0.5 onda to ukazuje da se lokacija ekstrema nalazi bliže nekoj drugoj interpoliranoj točki. Interpolacija se ponavlja oko nove točke  $\hat{x} = x + \hat{x}$ . Ako interpolacija konvergira nakon određenog broja iteracija, izračunati pomak  $\hat{x}$  se dodaje lokaciji i skali na kojoj je ekstrem originalno pronađen. Ilustraciju problema možemo vidjeti na slici 2.4.



Slika 2.4: Ilustracija problema interpolacije

4. **Odbacivanje nestabilnih značajki:** Algoritam koristi dvije metode za odbacivanje značajki.

(a) **Odbacivanje značajki na temelju niskog kontrasta:** Izračunati  $\hat{x}$  se koristi za odbacivanje značajki s lošim kontrastom koje imaju visoku osjetljivost na šum. Kombinacijom 2.15 i 2.16 dobivamo sljedeće:

$$D(\hat{x}) = D + \frac{1}{2} \frac{\partial D^T}{\partial x} \hat{x} \quad (2.17)$$

Prema [16] značajke koje imaju vrijednosti  $|D(\hat{x})| < 0.03$  se definiraju kao značajke s lošim kontrastom te se odbacuju.

- (b) **Odbacivanje značajki na temelju slabih rubova:** Česta pojava je da DoG algoritam značajke detektira na rubovima koji mogu biti slabo definirani i osjetljivi na šum. Takve značajke su nestabilne i potrebno ih je eliminirati. Loše definirani rubovi imaju veliku glavnu zakrivljenosti (*engl. principal curvature*) duž ruba a malu u ortogonalnom smjeru. Glavna zakrivljenost se računa pomoću  $2 \times 2$  Hessianove matrice  $H$  u točki značajke:

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}. \quad (2.18)$$

Vlastite vrijednosti Hessianove matrice  $H$  su proporcionalne glavnim zakrivljenostima parcijalnih derivacija u točki značajke. S obzirom na to da nas zanima samo njihov međusobni odnos, nećemo računati vlastite vrijednosti. Definiramo li sa  $\alpha$  vlastite vrijednosti najveće magnitude, a s  $\beta$  vlastite vrijednosti manje magnitude, tada sumu vlastitih vrijednosti možemo izračunati preko traga matrice  $H$ , a umnožak preko determinante:

$$Tr(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta \quad (2.19)$$

$$Det(\mathbf{H}) = D_{xx}D_{yy} - D_{xy}^2 = \alpha\beta \quad (2.20)$$

Ako je vrijednost determinante negativna, značajka se odbacuje. Označimo li s  $r$  odnos između vlastitih vrijednosti da vrijedi  $\alpha = r\beta$ , možemo napisati sljedeću jednadžbu:

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} = \frac{(\alpha + \beta)^2}{\alpha\beta} = \frac{(r\beta + \beta)^2}{r\beta^2} = \frac{(r + 1)^2}{r}. \quad (2.21)$$

Provjera da li je glavna zakrivljenost ispod nekog praga (*engl. threshold*)  $r$  računa se pomoću sljedećeg izraza:

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} < \frac{(r + 1)^2}{r}. \quad (2.22)$$

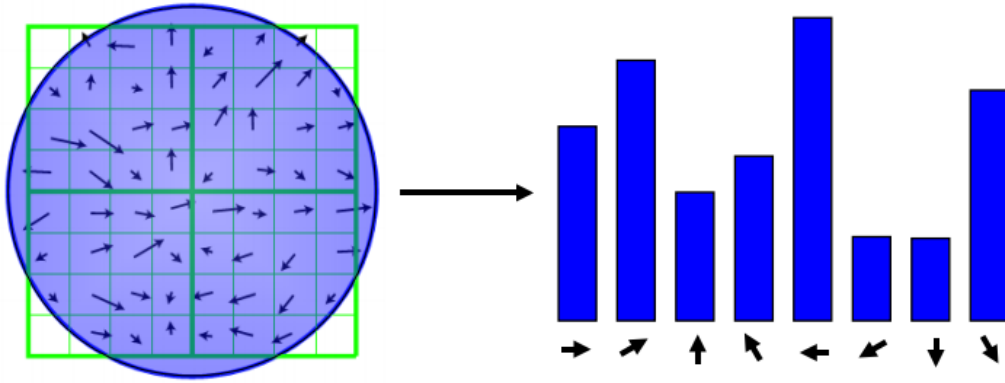
U [16] je preporučeni prag  $r = 10$ .

5. **Dodjeljivanje orijentacije:** Metoda koja se koristi predstavljena je u [20] u kojoj je svako svojstvo slike bazirano na mjerenjima invarijantnim na rotaciju. Nedostatak takvog pristupa je u tome što ograničava značajke koje mogu biti korištene. Za svaki slikovni element  $L(x, y)$  računa se gradijent magnitude  $m(x, y)$  i orijentacija  $\Theta(x, y)$  prema sljedećim izrazima:

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \quad (2.23)$$

$$\Theta(x, y) = \tan^{-1}((L(x+1, y) - L(x-1, y)) / (L(x, y+1) - L(x, y-1))) \quad (2.24)$$

Zatim pomoću izračunatih gradijenata možemo odrediti orijentacijski histogram koji se sastoji od 36 blokova koji prekrivaju 360° (svaki blok pokriva 10°). Svaka se izračunata orijentacija značajke dodaje u histogram i množi s njezinom izračunatom magnitudom i težinskim faktorom kojeg čini cirkularni Gaussov prozor širine  $\sigma$  veličine 1.5 puta veličina značajke.



**Slika 2.5:** Oblikovanje orijentacijskog histograma

Maksimumi u orijentacijskom histogramu odgovaraju dominantnim smjerovima gradijenata oko određene značajke. Ako u histogramu ima vrhova koji su minimalno 80% najvišeg vrha, na tim mjestima se kreiraju nove značajke s istim lokacijama i skalama ali različitim orijentacijama.

6. **Izgradnja deskriptora:** U prethodnim koracima svim značajkama su pridijeljene lokacije, skale i orijentacije pomoću kojih je postignuta invarijantnost na skalu i rotaciju. Cilj ovog koraka je postići invarijantnost na promjene u osvjetljenju i pomaku kamere.

Rješenje takvog problem zasniva se na [7], čiji pristup uključuje područje biološkog vida odnosno simulaciju kompleksnih neurona iz primarne vizualne moždane opne (*engl. primary visual cortex*). Neuroni odgovaraju gradijentu određene orijentacije i prostorne frekvencije, te lokacije koja je podložna malim promjenama. Male promjene u lokaciji su se pokazale kao dobro svojstvo za ostvarivanje invarijantnosti s obzirom na promjene položaja kamere.

Izgradnja deskriptora započinje gradijenata oko svake značajke. Slika u kojoj se određuju gradijenti odgovara skali značajke koja se trenutno promatra. Koordinate deskriptora i orijentacija gradijenta su rotirane relativno u odnosu na orijentaciju značajke iz razloga da se postigne invarijantnost na orijentaciju. Na magnitudu gradijenta primjenjuje Gaussov prozor širine  $\sigma$ . Time se gradijentima koji su udaljeniji od značajke daje manji značaj. Tavi gradijenti su najosjetljiviji na promjene. Nakon toga određujemo orijentacijski histogram od 8 blokova u  $4 \times 4$  susjedstvu oko značajke. Time se postiže da pomak gradijenta od ukupno 4 pozicije još uvijek pridonosi istom histogramu. Za izbjegavanje naglih promjena deskriptora uzrokovanih pomicanjem gradijenata, koristi se trilinearna interpolacija. Svaki uzorak histograma množi se težinskim faktorom  $1 - d$  za svaku dimenziju, gdje je  $d$  udaljenost uzorka od centralne vrijednosti bloka. Deskriptor se formira koristeći izračunati orijentacijske histograme.

Za izbjegavanje invarijantnosti na promjene u osvjetljenju, vrši se normalizacija. Time se poništavaju linearne promjene u kontrastu ili u osvjetljenju. Nelinearne promjene osvjetljenja pridonose velike promjene u magnitudi gradijenta. Iz tog razloga se postavlja prag na magnitude koji iznosi 0.2.

### 2.2.5. SURF (*engl. speeded up robust features*)

Detektor značajki temeljen na Hassian-Laplaceovom detektoru invarijantan na skale predložen je u [3] i [4]. Kako bi se izbjeglo direktno računanje Hassianove matrice koristi se aproksimacija pomoću *integralne slike* (*engl. integral images*). Integralne slike omogućuju vrlo brz izračun Haarovih veličina (*engl. Haar wavelets*), ali i bilo kojeg konvolucijskog filtra koji se temelji na usrednjavanju susjednih elemenata (*engl. box-type convolution filter*). Integralna slika  $I_{\Sigma}(x, y)$  definirana je sljedećim izrazom:

$$I_{\Sigma}(x, y) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(x, y) \quad (2.25)$$

Osim aproksimacije pomoću integralnih slika, u ovom detektoru koristi se i aproksimacija pomoću filtra s maskama za usrednjavanje. Aproksimacija determinante računa se pomoću sljedećeg izraza:

$$\det(H_{approx}) = D_{xx}D_{yy} + (0.6D_{xy})^2 \quad (2.26)$$

gdje su  $D_{xx}, D_{xy}, D_{yy}$  aproksimacijski filtri a težinska vrijednost je 0.6. Vrijednost determinante predstavlja odziv grudaste značajke na lokaciji  $x$ .



## 3. Lokalizacija kamere u prostoru

Estimacija položaja i pokreta stereo kamere jedna je od glavnih značajki SLAM-a. Stereo kamera se u estimaciji pozicije autonomne podvodne ronilice koristi iz razloga jer nije moguće koristiti GPS tehnologiju u podvodnim sustavima a jeftinija je od preciznih IMU (*Inertial measurement unit*) sustava. Takav pristup određivanja pomaka temeljen na vizualnim podacima naziva se *vizualna odometrija* (engl. *visual odometry*) [18]. Prije samog početka korištenja stereo kamere u određivanju odometrije, stereo kameru je potrebno umjeriti odnosno kalibrirati kako bi dobili što preciznije podatke u određivanju vizualne odometrije [12].

### 3.1. Model kamere

U ovom radu korištena je stereo kamera Bumblebee XB3 koja se sastoji od 3 monokromatske kamere razmaknute za fiksne udaljenosti. Razlog korištenja 3 kamere umjesto običnog stereo sustava koji se sastoji od dvije kamere razmaknute na fiksnoj udaljenosti je u dodatnoj preciznosti određivanja dubine scene koja će biti kasnije objašnjena. U radu su korištene samo dvije kamere, odnosno nije korištena centralna kamera jer je pretpostavka da se promatra scena na većim udaljenostima gdje kamere na većoj međusobnoj udaljenosti daju bolje rezultate. Specifikacije stereo sustava kamera su prikazane u tablici 3.1, a sama kamera je prikazana na slici 3.1.

Kako bi lokalizacija kamere bila uspješna potrebno je poznavati matematički model kamere. Kamerom projiciramo točke iz 3D prostora u 2D sliku te je matematički možemo opisati na sljedeći način:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = Z \begin{bmatrix} \frac{fX}{Z} \\ \frac{fY}{Z} \\ 1 \end{bmatrix} \quad (3.1)$$

Resolution	1208x960
Frame Rate	16 FPS
Megapixels	1.3 MP
Chroma	Mono
Sensor name	Sony ICX445
Sensor Type	CCD
Pixel Size	3.75 $\mu\text{m}$
Focal Length	6 mm, 43-deg HFOV
ADC	12-bit
Exposure Range	0.03 ms to 66.63 ms
Flash Memory	512 KB non-volatile memory
Serial Port	2 RS-232
Interface	FireWire 1394b
Power Requirements	12 V
Dimensions	277 m x 37 mm x 41.8 mm
Mass	505 grams

**Tablica 3.1:** Specifikacije kamere Bumblebee XB3



**Slika 3.1:** Bumblebee XB3

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{fX}{Z} \\ \frac{fY}{Z} \\ 1 \end{bmatrix} \quad (3.2)$$

gdje su  $u$  i  $v$  koordinate točaka u koordinatnom sustavu slike,  $s$  je informacija o dubini scene koja se gubi projekcijom,  $f$  je žarišna duljina ili udaljenost ravnine slike od

centra projekcije, a  $X, Y$  i  $Z$  su koordinate promatrane točke u 3D prostoru. Jednadžba 3.1 prikazuje perspektivnu projekciju točke  $\mathbf{P} = (X, Y, Z)^T$  iz 3D prostora u točku slikovne ravnine  $x = (fX/Z, fY/Z)$ . Dana jednadžba vrijedi za model kamere kojoj centralna os prolazi kroz središte ravnine slike. Model kamere koji se u praksi koristi ima centralnu os u gornjem lijevom kutu, odnosno na slikovnom elementu  $(0,0)$ . Model kamere je tada definiran na sljedeći način:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = Z \begin{bmatrix} \frac{fX}{Z} + c_x \\ \frac{fY}{Z} + c_y \\ 1 \end{bmatrix} \quad (3.3)$$

. Uveli smo translacijske parametre  $c_x$  i  $c_y$  kako bi se gornji lijevi slikovni element našao na centralnoj osi.

Model je potrebno proširiti tako da se omogući gibanje scene u odnosu na kameru tako da uvodimo transformacijsku matricu  $\mathbf{Rt}$  rotaciju i translaciju točaka prije njihovog projiciranja na ravninu slike.

$$\mathbf{Rt} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.4)$$

Model kamere sada možemo zapisati kao:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (3.5)$$

. Ako želimo modelirati kretanje kamere s obzirom na promatranu scenu, što je potrebno kod lokalizacije kamere u prostoru, na mjesto  $\mathbf{Rt}$  je potrebno upisati inverznu transformaciju gibanja kamere. Budući da kameru promatramo u centru fiksnog koordinatnog sustava, promatrane točke scene je potrebno u položaj u kojem bi se našle ako bi se kamera gibala sa svojim koordinatnim sustavom.

Prikazani model je pojednostavljeni model realne kamere. Jedan od problema koji se javljaju kod realne kamere je distorzija leće. Leće koriste kako bi dobili oštrinu slike na način da usmjeravaju zrake svjetlosti u uzak snop i projicira ih na površinu slike. Upravo zato što ovo svojstvo nije konstantno i ovisi o mjestu upada zrake na

leću dolazi do problema koji se manifestira kao efekt radijalne distorzije ili izobličenja slike. Distorzija se javlja na rubovima leće te je i izobličenje vidljivo na rubovima slike.

Drugi problem je uzrokovan zbog nemogućnosti idealnog poravnanja leće i foto senzora kamere koji nisu savršeno paralelni. Neidealno poravnanje uzrokuje tangencijalnu distorziju.

U pravilu je učinak radijalne distorzije puno više izražen od učinka tangencijalne distorzije, radijalna distorzija se modelira s većim brojem parametara. Izobličene koordinate slike  $(x_d, y_d)$  se definiraju preko neizobličenih koordinata  $(x_u, y_u)$  na sljedeći način:

$$x_d = x_u(1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6) + dx \quad (3.6)$$

$$y_d = y_u(1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6) + dy \quad (3.7)$$

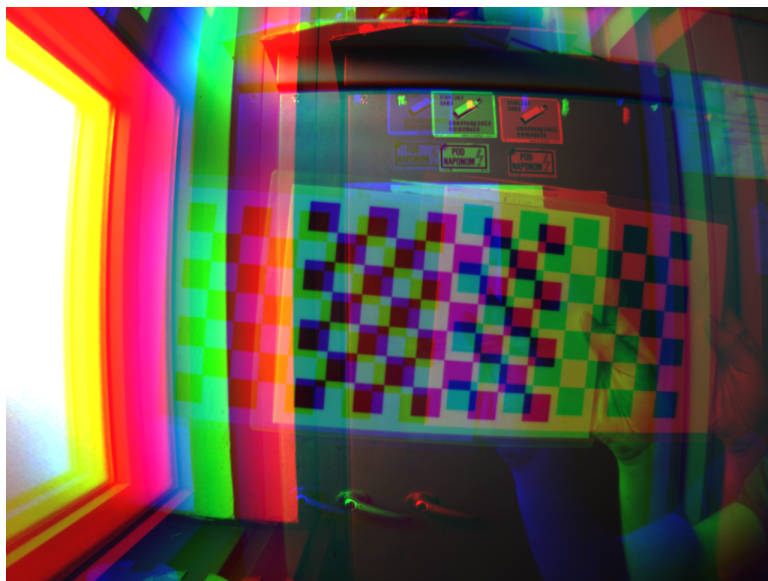
$$r^2 = x_u^2 + y_u^2 \quad (3.8)$$

gdje su parametri tangencijalne distorzije  $dx$  i  $dy$  definirani na sljedeći način:

$$dx = 2\tau_1 x_u y_u + \tau_2 (r^2 + 2x_u^2) \quad (3.9)$$

$$dy = 2\tau_2 x_u y_u + \tau_1 (r^2 + 2y_u^2). \quad (3.10)$$

Izvorna slika koju dobivamo s kamere prikazana je na slici 3.2, te je sliku potrebno programski razdijeliti na zasebne monokromatske prikaze scene kako bi dobili stereo par slika.



**Slika 3.2:** Izvorna slika dobivena sa sustava kamera Bumblebee XB3

### 3.1.1. Kalibracija stereo kamere

Kalibracija stereo sustava je postupak kojim određujem unutarnju geometriju kamere koja utječe na proces stvaranja digitalne slike. Parametre koje određujemo kalibracijom su:

1. Intrinzični parametri:

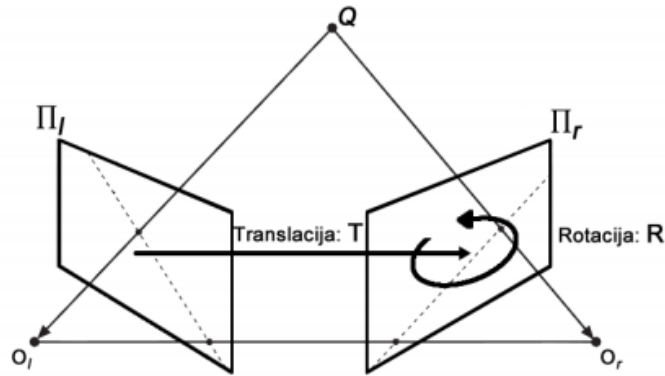
- (a) Žarišna duljina, udaljenost slikovne ravnine od projekcije.
- (b) Koeficijent ukošenosti kojim se modelira nejednaka horizontalna i vertikalna udaljenost slikovnih elemenata.
- (c) Radijalna ( $\kappa_1, \kappa_2, \kappa_3$ ) i tangencijalna ( $\tau_1, \tau_2$ ) distorzija leće kojima se opisuju promjene u slici nastale zbog nesavršenosti optičkih leća.

Matrica koja sadrži intrinzične parametre naziva se matrica kamere  $\mathbf{K}$  i glasi:

$$\mathbf{K} = \begin{bmatrix} f & 0 & c_u & 0 \\ 0 & f & c_v & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (3.11)$$

2. Ekstrinzični parametri: određuje se međusobni položaj dvije kamere odnosno rotacijska  $\mathbf{R}$  i translacijska  $\mathbf{T}$  matrica.

Kalibracijom je potrebno odrediti ekstrinzične parametre koji su prikazani na slici 3.3. Pri kalibraciji prostorna točka  $Q$  se prikazuje u koordinatnom sustavu lijeve kamere



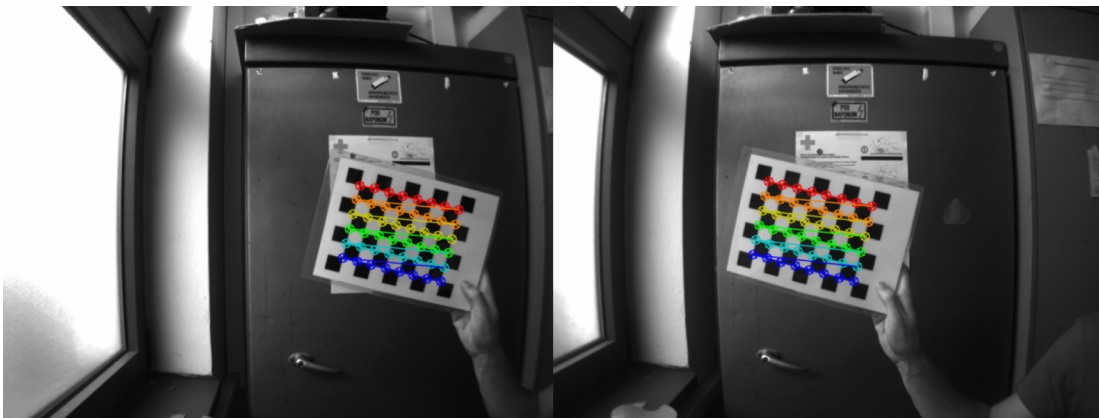
**Slika 3.3:** Ekstrinzični parametri stereo sustava

kao  $Q_l = R_l Q + T_l$ , odnosno u koordinatnom sustavu desne kamere kao  $Q_r = R_r Q + T_r$ . Dalje vrijedi  $Q_l = R^T(Q_r - T)$ . Konačni izrazi za rotaciju i translaciju su sljedeći:

$$R = R_r(R_l)^T \quad (3.12)$$

$$T = T_r - RT_l \quad (3.13)$$

Kamera se kalibrira na način da se promatra neki objekt čija je geometrija poznata. Najčešće se u tu svrhu koristi asimetrična šahovska ploča. Algoritmu za kalibraciju potrebno je dati duljinu stranice kvadrata šahovske ploče. Kalibracija će biti to bolja što je snimljeno više kalibracijskih uzoraka pod raznim kutevima i na raznim udaljenostima. Kalibracija stereo kamere i detektirani rubovi asimetrične šahovske ploče koja se koristi u postupku kalibracije prikazani su na slici 3.4. Asimetričnost ploče se koristi iz razloga mogućnosti prepoznavanja rotacije ploče s obzirom na koordinatni sustav kamere.



Slika 3.4: Postupak kalibracije stereo kamere

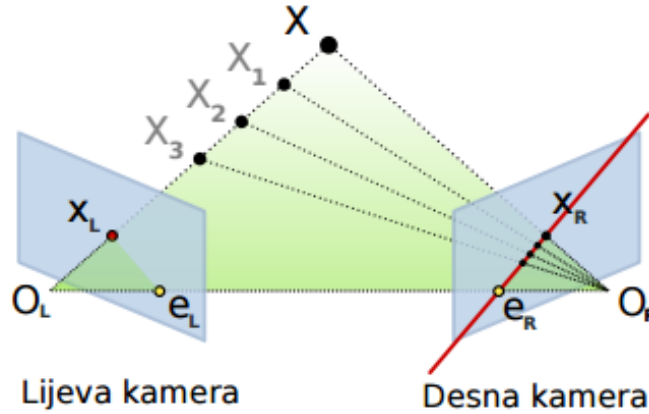
### 3.1.2. Epipolarna geometrija

Geometriju sustava stereo kamera možemo opisati epipolarnom geometrijom. Epipolarna geometrija je prikazana na slici 3.5. Prikazane kamere gledaju u smjeru točke  $X$  u sceni. S  $O_L$  i  $O_R$  su označeni centri projekcije na lijevoj i desnoj kameri.  $x_L$  i  $x_R$  su projekcije točke  $X$  na ravninu slika. Točke  $e_L$  i  $e_R$  se nazivaju epipolovima i definiraju se kao sjecišta pravaca određenih točkama  $O_L$  i  $O_R$  te lijeve i desne ravnine projekcije. Projekcija pravca  $O_L - X$  na desnu kameru definira epipolarni pravac  $e_R - x_R$ , dok projekcija pravca  $O_R - X$  na lijevu kameru definira epipolarni pravac  $e_L - x_L$ .

Ako su poznate relativna translacija i rotacija između dvije kamere u stereo sustavu, pomoću epipolarne geometrije možemo doći do dva bitna opažanja:

1. Ako je poznata točka projekcije  $x_L$  tada je poznat i epipolarni pravac  $e_R - x_R$ . Točka  $X$  tada projicira na točku  $x_R$  u desnoj slici koja mora ležati na spomenutom epipolarnom pravcu.

2. Ako su nam poznate točke projekcije  $x_L$  i  $x_R$ , poznati su nam i njihovi projekcijski pravci koji se sijeku točno u  $X$ . Iz poznavanja pozicija projekcija točke  $X$  na lijevu i desnu kameru moguće je odrediti njene koordinate u 3D prostoru. Postupak je opisan kasnije i naziva se triangulacija.



Slika 3.5: Epipolarna geometrija stereo para kamera

### 3.1.3. Esencijalna i fundamentalna matrica

Ideja određivanja epipolarnih pravaca je da se korespondentne značajke nalaze na istim pravcima. Postavljanje korespondentnih značajki na iste pravce naziva se epipolarnim ispravljanjem. Za matematički opis epipolarnog ispravljanja potrebno je uvesti pojmove esencijalne i fundamentalne matrice.

Esencijalna matrica  $\mathbf{E}$  sadrži informacije o translaciji i rotaciji između dvije kamere u stereo sustavu. Fundamentalna matrica  $\mathbf{F}$  osim informacija koje sadrži esencijalna matrica, sadrži dodatno informacije o intrinzičnim parametrima dviju kamera.

Ako imamo točku  $\mathbf{P}$  iz 3D prostora s projekcijom na koordinatni sustav prve kamere  $\mathbf{x}_1 = (x_1, y_1, z_1)$  i projekcijom na koordinatni sustav druge kamere  $\mathbf{x}_2 = (x_2, y_2, z_2)$ . Ako su kamere normalizirane, korespondentne koordinate točke u lijevoj i desnoj slici nakon perspektivne projekcije su:

$$\mathbf{y}_1 = \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = \frac{1}{z_1} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \quad (3.14)$$

$$\mathbf{y}_2 = \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix} = \frac{1}{z_2} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} \quad (3.15)$$

Budući da su koordinatni sustavi normaliziranih kamera u relaciji preko translacijske i rotacijske matrice, koordinate točaka se iz jedne projekcije mogu lako prebaciti u drugu projekciju na sljedeći način:

$$\mathbf{x}_2 = \mathbf{R}(\mathbf{x}_1 - \mathbf{t}) \quad (3.16)$$

Esencijalnu matricu možemo tada definirati kao:

$$\mathbf{E} = \mathbf{R}[\mathbf{t}]_x \quad (3.17)$$

gdje je  $[\mathbf{t}]_x$  matrična reprezentacija vektorskog produkta s vektorom  $\mathbf{t}$  i detaljnije je prikazana izrazom 3.18:

$$\mathbf{t} \times \mathbf{a} = [\mathbf{t}]_x \mathbf{a} = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad (3.18)$$

Kako esencijalna matrica sadrži samo informacije o geometriji jedne kamere u odnosu na drugu a ne sadrži nikakve informacije o samim kamerama potrebno je definirati fundamentalnu matricu. Kako bi dobili fundamentalnu matricu, točku  $\mathbf{y}$  u normaliziranim koordinatama supstituiramo točkom  $\mathbf{y}'$  u koordinatnom sustavu kamere s intrinzičnim parametrima  $\mathbf{K}$  i vrijedi sljedeće:

$$\mathbf{y}' = \mathbf{K}\mathbf{y} \quad (3.19)$$

$$\mathbf{y}'_2{}^T (\mathbf{K}_2^{-1})^T \mathbf{E} \mathbf{K}_1^{-1} \mathbf{y}'_1 = 0 \quad (3.20)$$

$$\mathbf{F} = (\mathbf{K}_2^{-1})^T \mathbf{E} \mathbf{K}_1^{-1} \quad (3.21)$$

Jednadžbom 3.21 definirali smo fundamentalnu matricu i sada možemo izraz 3.20 zapisati kao:

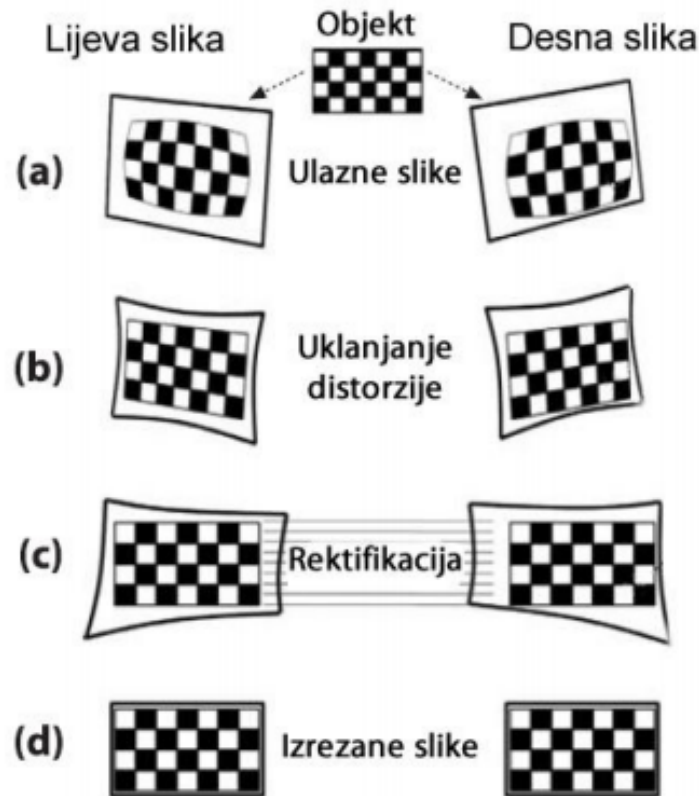
$$\mathbf{y}'_2{}^T \mathbf{F} \mathbf{y}'_1 = 0 \quad (3.22)$$

### 3.1.4. Rektifikacija stereo para slika

Rektifikacija stereo para slika je proces ispravljanja lijeve i desne slike dobivene iz stereo kamere tako da budu poravnate po recima te da optičke osi njihovih virtualnih



kamera budu paralelne (sijeku se u beskonačnosti). Razlog provedbe postupka rektifikacije je taj da se pretraživanje korespondentnih točaka svede na pretraživanje redaka te se time ubrza proces njihovog pronalaženja i povećava se točnost [9]. Proces rektifikacije prikazan je na slici 3.6.



**Slika 3.6:** Postupak rektifikacije stereo para

Nakon rektifikacije stereo para, intrinzični parametri lijeve i desne kamere su u potpunosti jednaki. Model rektificiranog sustava možemo zapisati u obliku:

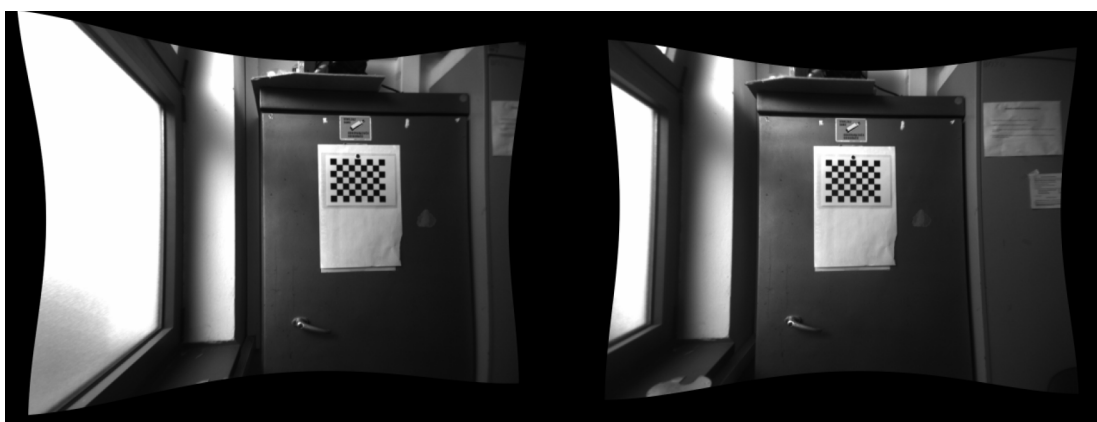
$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & c_x & 0 \\ 0 & f & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \left( \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} - \begin{bmatrix} t_b \\ 0 \\ 0 \\ 0 \end{bmatrix} \right) \quad (3.23)$$

gdje je  $t_b$  udaljenost između kamera odnosno vrijednost za lijevu kameru je  $t_b = 0$  a za desnu  $t_b = b$ , *engl. baseline*.

Na slikama 3.7 - 3.9 je prikazan proces rektifikacije para stereo slika, te je vidljivo kako je uklonjena distorzija leće koja se najviše očituje na rubovima slika. Osim rektifikacije slike su i epipolarno poravnate tako da se značajke nalaze na istim epipolarnim linijama, što se vidi na slici 3.9.



**Slika 3.7:** Ulazni par stereo slika



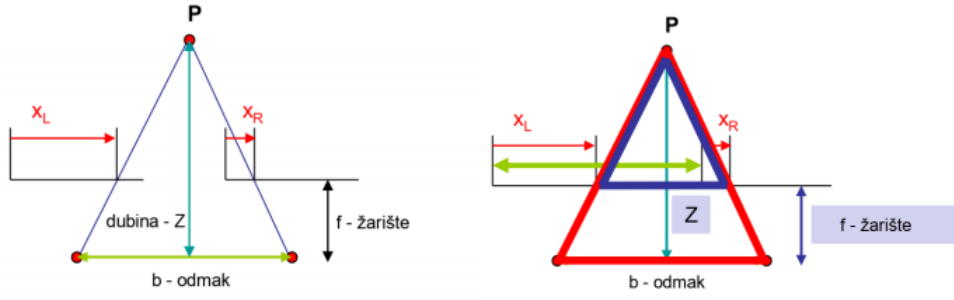
**Slika 3.8:** Rektificirani par stereo slika s uklonjenom distorzijom leće



**Slika 3.9:** Izrezani dio para stereo slike koji se koristi u daljnjem procesu

### **3.1.5. Triangulacija i matrica dispariteta**

Nakon rektifikacije stereo sustava, iz stereo para slika možemo odrediti dubinu promatrane scene. Postupak određivanja dubine scene naziva se triangulacija i prikazan je na slici 3.10.



**Slika 3.10:** Postupak rektifikacije stereo para

Pomoću sličnosti trokuta označenim crvenom i plavom bojom dobivamo dubinu  $Z$ , odnosno 3D koordinate točke  $P$  u prostoru. Jednadžbe za određivanje pozicije točke  $P$  dane su u nastavku:

$$P = [X, Y, Z]^T \quad (3.24)$$

$$\frac{b}{Z} = \frac{b + x_R - x_L}{Z - f}; Z = \frac{f \cdot b}{x_L - x_R} = \frac{f \cdot b}{d} \quad (3.25)$$

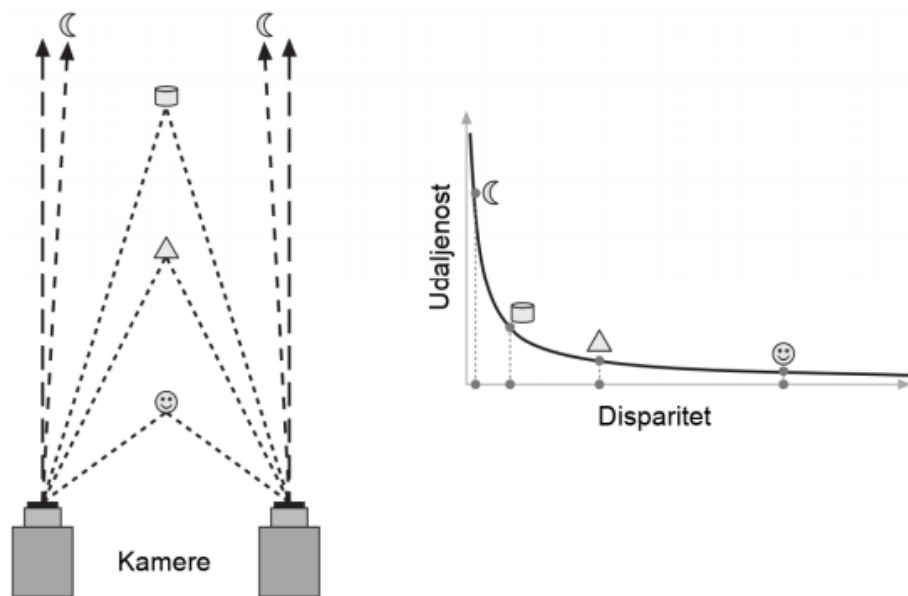
$$\frac{X}{Z} = \frac{x_L}{f}; X = \frac{x_L}{f} \cdot \frac{f \cdot b}{d} = \frac{x_L \cdot b}{d} \quad (3.26)$$

$$\frac{Y}{Z} = \frac{y_L}{f}; Y = \frac{y_L}{f} \cdot \frac{f \cdot b}{d} = \frac{y_L \cdot b}{d} \quad (3.27)$$

gdje  $f$  označava fokalnu duljinu,  $d$  je disparitet, a  $b$  je udaljenost između lijeve i desne kamere.

Bitno svojstvo kod diskretizacije perspektivne projekcije koje u realnoj kameri obavlja senzor jest da vrijednosti dispariteta nelinearno ovise o trianguliranoj udaljenosti tj. za manje vrijednosti dispariteta udaljenost triangulirane značajke eksponencijalno raste što možemo vidjeti na slici 3.11. To svojstvo perspektivne projekcije ne možemo izbjeći niti na neki način kompenzirati. Bliži objekti se kod perspektivne projekcije projiciraju u lijevu i desnu sliku s većim disparitetom te je informacija o njihovoj dubini sačuvana uz manje šuma, dok za objekte na velikim udaljenostima nekad nije moguće odrediti disparitet, a samim time i udaljenost objekta od stereo kamere. Za udaljenije objekte veću preciznost dobivamo ako su kamere međusobno više udaljene.

Ako sliku prikažemo kao skup slikovnih elemenata čija se nijansa određuje na temelju udaljenosti od stereo kamere dobivamo matricu dispariteta. Praksa je da se bliže točke u prostoru označe svjetlijim nijansama slikovnih elemenata, a udaljenije točke označe tamnijim nijansama slikovnih elemenata. Matricu dispariteta možemo vidjeti na slici 3.12 u donjem desnom kutu. Za određivanje matrice dispariteta koristimo sljedeće parametre:



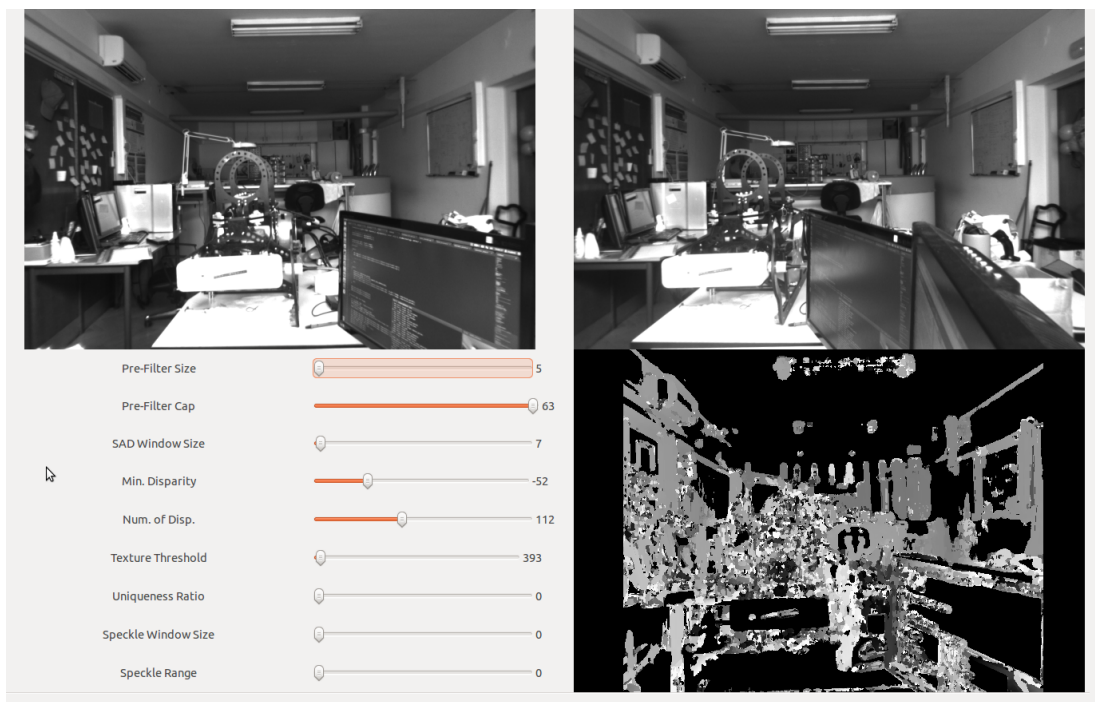
**Slika 3.11:** Ovisnost dispariteta o udaljenosti scene

- minDisparity: Minimalna vrijednost dispariteta.
- numDisparities: Razlika između maksimalnog i minimalnog dispariteta.
- SADWindowSize: Veličina bloka za uspoređivanje lijeve i desne slike.
- preFilterCap: Računa se x-derivat svakog piksela te se zatim vrijednost skalira na interval  $[-\text{preFilterCap}, \text{preFilterCap}]$ . Rezultantne vrijednosti su proslijeđene *Birchfield-Tomasi* vrijednosnoj funkciji piksela [5].
- uniquenessRatio: Postotak kojim minimalna cijena piksela "pobjeđuje" drugu minimalnu cijenu piksela da se smatra točnim uparivanjem.
- speckleWindowSize: Maksimalna veličina prozora koji se koristi za zaglađivanje dispariteta.
- speckleRange: Maksimalna razlika između dispariteta susjednih područja.

## 3.2. Vizualna odometrija

### 3.2.1. Pronalaženje korespondentnih značajki

U određivanju vizualne odometrije korištena je biblioteka *Libviso2* (*Library for Visual Odometry 2*) [11] pomoću koje se određuje šest stupnjeva slobode koristeći potpuno



**Slika 3.12:** Par stereo slika (gore), parametri za definiranje matrice dispariteta (dolje lijevo), matrica dispariteta (dolje desno)

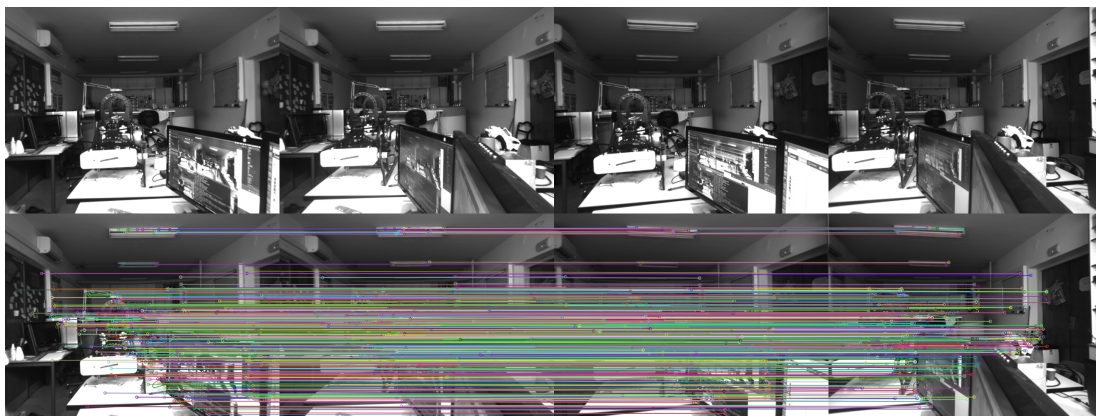
kalibrirani par stereo kamera. Određivanje vizualne odometrije provodi se u stvarnom vremenu.

Na ulaz u algoritam šalju se značajke dobivene podudaranjem između četiriri slike, trenutne lijeve i desne te prethodne lijeve i desne prikazane na slici 3.14.



**Slika 3.13:** Korespondentne značajke između para stereo slika

Za pronalazak stabilnih značajki koristi se već prethodno opisani SURF algoritam. Detektirane značajke u trenutno lijevoj slici se uspoređuju i uparuju s detektiranim značajkama u trenutno desnoj slici. Uparene značajke se zatim uspoređuju s uparenim



**Slika 3.14:** Usporedba značajki između prethodnog stereo para (lijevi skup slika) i trenutnog stereo para (desni skup slika)

značajkama u prethodno lijevo i desnoj slici. Korespondentne značajke čiji disparitet ili optički tok prelazi granicu  $\tau_{disp}$  i  $\tau_{flow}$  se odbacuju.

Iz velikog broja detektiranih značajki u svakoj iteraciji, uniformno se selektiraju značajke s minimalnim ili maksimalnim odzivom tako da se slika podijeli na više pravokutnih područja te se u svakom području odredi fiksni broj značajki. Ovakav pristup nije uvijek najbolji iz razloga jer je moguće da se odaberu jako loše značajke u područjima gdje je mali broj ukupno detektiranih značajki.

### 3.2.2. Estimacija gibanja

U estimaciji pokreta računa se pozicija kamere na temelju slika snimljenih u trenutnom vremenskom okviru i slika snimljenih u neposredno prethodnom vremenskom okviru. Povezivanjem svih pokreta u lanac dobivamo trajektoriju gibanja kamere. Estimacija pokreta može se računati na tri različita način obzirom ako je korespondencija značajki definirana u dvije ili tri dimenzije [6]:

- 2D-to-2D: U ovom slučaju korespondentne značajke prethodnog i trenutnog vremenskog okvira su definirane u 2D koordinatama slika.
- 3D-to-3D: Korespondentne značajke su definirane u 3D prostoru. Za korištenje ove metode potreban je korak triangulacije koji je prethodno objašnjen.
- 3D-to-2D: Značajke u prethodnom vremenskom okviru su definirane u 3D prostoru, dok su trenutne značajke definirane u 2D koordinatama slike.

U ovom je radu korištena 3D-to-3D metoda s obzirom da se koristi stereo kamera kako bi se odredila dubina scene.

### 3D-to-3D estimacija pokreta

Estimaciju pokreta kamere između prethodnog i trenutnog video okvira možemo definirati kao:

$$T_k = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix}. \quad (3.28)$$

$T_k$  računamo iz korespondentnih značajki susjednih vremenskih okvira snimljenih u vremenu  $t_k - 1$  i  $t_k$ , te je potrebno odrediti takav  $T_k$  da minimizira  $L_2$  udaljenost između dva seta značajki dobivenih u tim vremenskim intervalima na sljedeći način:

$$\arg \min_{T_k} \sum_i \|\tilde{X}_k^i - T_k \tilde{X}_{k-1}^i\| \quad (3.29)$$

gdje su  $\tilde{X}_k$  i  $\tilde{X}_{k-1}$  homogene koordinate 3D točaka u obliku  $\tilde{X} = [x, y, z, 1]^T$ . U [2] je prikazano kako slučaj za minimalan rezultat uključuje tri 3D-to-3D nekolinearne korespondencije. Translaciju računamo kao razliku centroida skupa 3D značajki, dok rotaciju računamo korištenjem SVD algoritma [14].

Translacija se dobiva kao:

$$t_k = \bar{X}_k - R\bar{X}_{k-1} \quad (3.30)$$

gdje  $\bar{\cdot}$  označava aritmetičku sredinu.

Rotacija se računa korištenjem SVD algoritma kao:

$$R_k = VU^T \quad (3.31)$$

gdje je  $UV^T = \text{svd}((X_{k-1} - \bar{X}_{k-1})(X_k - \bar{X}_k)^T)$ , a  $X_{k-1}$  i  $X_k$  su setovi korespondentnih značajki.



## 4. Eksperimentalni rezultati

### 4.1. Rezultati zatvaranja petlje

Testiranje zatvaranja petlje provedeno je na tri različita skupa podataka snimljena različitim kamerama. Korišteni su sljedeći skupovi podataka:

1. Skup podataka rute hodnicima Fakulteta elektrotehnike i računarstva. Korištena razlučivost snimke je 384x216 piksela. Skup podataka sadrži 306 video okvira snimanih frekvencijom od jednog video okvira u sekundi.
2. Dva javna skupa podataka koji prikazuju zatvoreni prostor (*Lip6Indoor*) od 388 video okvira i vanjski prostor (*Lip6Outdoor*) od 531 video okvira razlučivosti 240x192 piksela [1]. Frekvencija snimanja je ista kao i u prethodnim skupovima podataka.

Formule koje se koristi za izračunavanje performanse algoritma su sljedeće:

$$Precision = \left( \frac{TP}{TP + FP} \right) \quad (4.1)$$

$$Recall = \left( \frac{TP}{TP + FN} \right) \quad (4.2)$$

gdje  $TP$  označava broj točno prihvaćenih slika za zatvaranje petlje,  $FP$  je broj slika koje su pogrešno prihvaćene u postupku određivanja zatvaranja petlje,  $FN$  je broj slika koje su pogrešno odbačene. Poželjna je što veća vrijednost varijable *Recall* što znači da je veliki broj mogućih zatvaranja petlje detektirano. Vrijednost *Precision* = 100% nam govori da nije detektirano i prihvaćeno niti jedno pogrešno zatvaranje petlje.

Za svaki slup podataka provedeni su sljedeći koraci procjene zatvorenosti petlje:

1. Definirani su SIFT parametri za izdvajanje značajki. Svaki video okvir sekvence je korišten za usporedbu s prethodnima kao video okvirima kako bi se detektiralo zatvaranje petlje ali nije uspoređivan s prethodno susjednih 10 video okvira i sam sa sobom.



2. Za svaki trenutni video okvir određuje se 5 najboljih kandidata za procjenu zatvaranja petlje.
3. Kada je potvrđeno zatvaranje petlje, vrši se usporedba s *ground truth* podacima. Ako u *ground truth* datoteci nije oznaka zatvaranja petlje, video okvir se definira kao *FP*.
4. Završno se svi video okviri koji nisu detektirani algoritmom ali u *ground truth* datoteci su označeni kao zatvaranje petlje definiraju kao *FN*.

Rezultati za sva tri skupa podataka prikazani su u tablici 4.1. Rezultati prikazuju da

Skup ispitnih podataka	Broj video okvira	Razlučivost	Vrijeme izvršavanja algoritma	<i>Precision</i>	<i>Recall</i>
FER	307	384x216	30.02s	100%	95.32%
Lip6Indoor	388	240x192	19.23s	100%	74%
Lip6Outdoor	1063	240x192	129.59s	100%	76%

**Tablica 4.1:** Rezultati detekcije zatvaranja petlje za različite skupove podataka

preciznost algoritma ne ovisi samo o ulaznim parametrima nego i o razlučivosti ulazne slike. Ako imamo veću razlučivost, detekcija zatvaranja petlje je bolja iz razloga jer je moguće izvući više značajki iz slike na temelju kojih se kasnije detektira zatvorenost petlje. Osim preciznosti, razlučivost uzrokuje i dulje vrijeme izvršavanja algoritma.

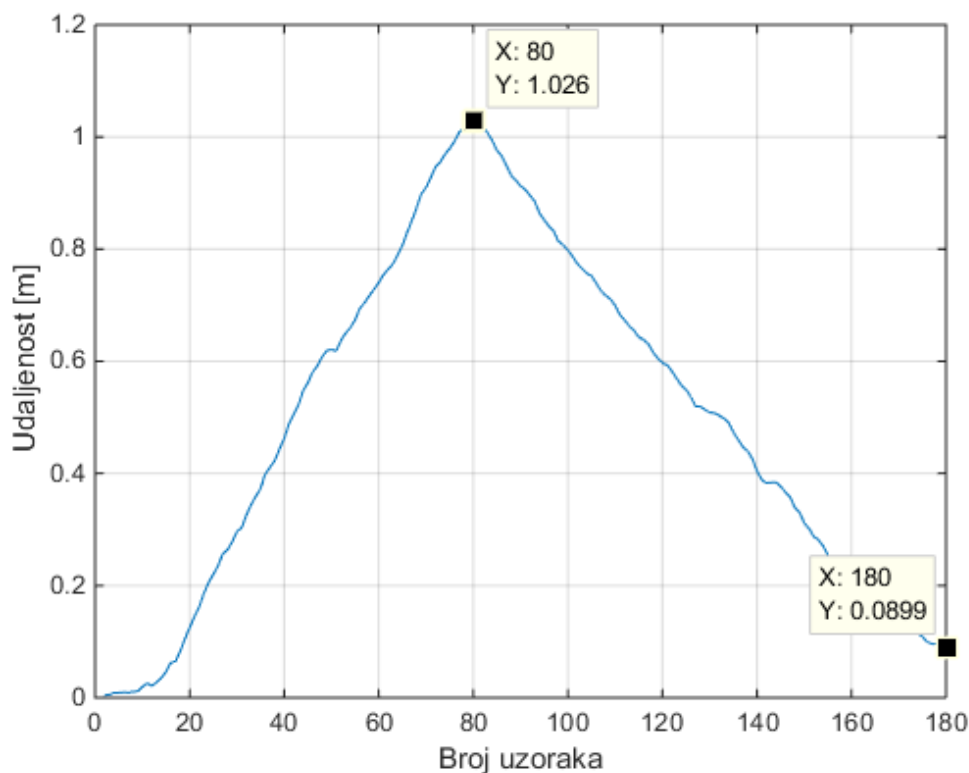
Vrijeme izvršavanja algoritma vrlo je bitno za sustave koji rade u stvarnom vremenu. Iz prikazanih rezultata u tablici 4.1 vidimo da je za prvi skup ispitnih podataka potrebno približno 0.1s za obradu pojedinog video okvira. Kod drugog skupa vrijeme je nešto manje i iznosi 0.04s, dok je za treći skup podataka vrijeme 0.12s.

Algoritam je uspješno testiran i u stvarnom vremenu na kameri Bumblebee XB3 koja koristi *frame rate* od 8.5 Hz, odnosno snima 8.5 video okvira u jednoj sekundi. Prije provedbe algoritma, sliku je potrebno smanjiti za 70% kako bi bilo moguće nesmetano izvršavanje na osobnom računalu.

## 4.2. Rezultati lokalizacije kamere u prostoru

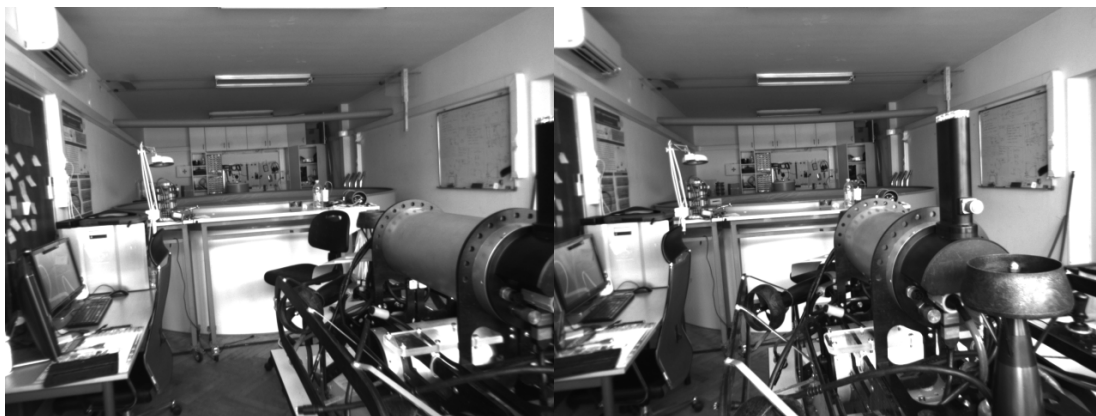
Napredovanje kamere testirano je pomoću mjerila od jednog metra. Kamera je pomicala duž mjerila te je nakon toga vraćena na početnu poziciju. Rezultati određivanja pomaka prikazani su na slici 4.1.

Na udaljenosti od jednog metra na mjerilu postoji pogreška od 26mm. Pri vraćanju kamere, pogreška se akumulira i krajnja pogreška iznosi približno 90mm. Scena iz



**Slika 4.1:** Prikaz pomaka stereo kamere po z osi (napredovanje), prvi eksperiment

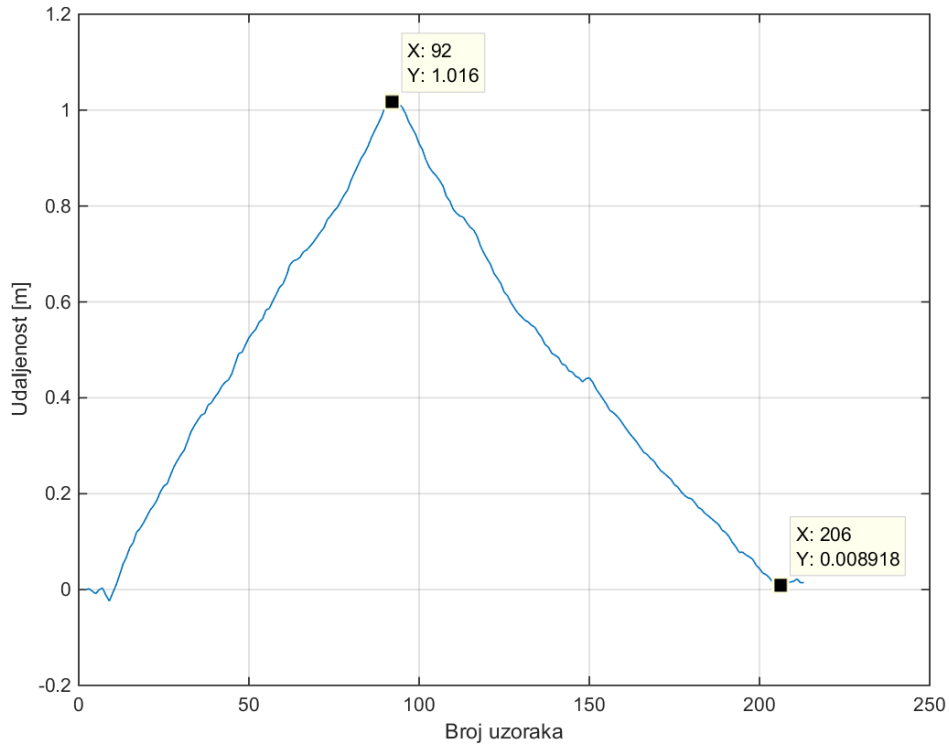
koje su izvlačene značajke prikazana je na slici 4.2. Vidimo da u sceni postoje bliži i dalji elementi te je pomak estimiran na temelju značajki tih elemenata.



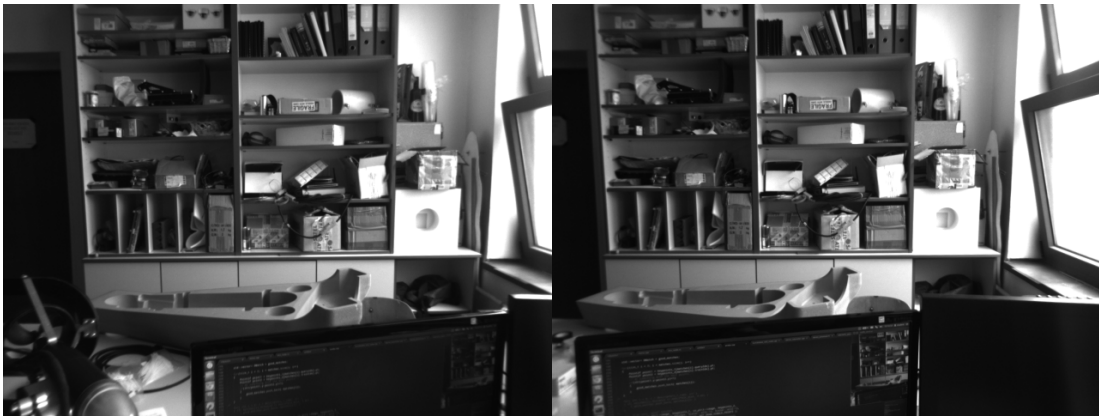
**Slika 4.2:** Scena promatrana pri eksperimentu napredovanja, prvi eksperiment

U drugom mjerenju korištene su značajke iz 3D prostora koje su na manjoj udaljenosti te su svi elementi u sceni relativno jednako udaljeni. Scena u ovom dijelu eksperimenta prikazana je na slici 4.4, a na slici 4.3 vidimo rezultate koji pokazuju nešto veću preciznost od prethodnog eksperimenta. Odstupanje na pomaku za jedan

metar je 16mm dok je pri vraćanju kamere u početnu poziciju odstupanje približno 9mm. Iz ovog eksperimenta možemo zaključiti da nema akumulacije pogreške.



**Slika 4.3:** Prikaz pomaka stereo kamere po z osi (napredovanje), drugi eksperiment



**Slika 4.4:** Scena promatrana pri eksperimentu napredovanja, drugi eksperiment

Možemo zaključiti da scena na kojoj je većina značajki relativno na jednakim udaljenostima daje bolje rezultate u određivanju odometrije kamere.

Pri mjerenju kuteva *Roll* i *Pitch* rezultati su uspoređeni s IMU senzorom *MicroS-train 3DM-GX3-25*. Sustav koji je korišten pri mjerenju prikazan je na slici 4.5. Rota-



**Slika 4.5:** Sustav stereo kamere i IMU-a korišten pri mjerenju *Roll* i *Pitch* kuta zakreta

cija *Roll* i *Pitch* na IMU-u dobiva iz akceleracije pomoću sljedećih formula:

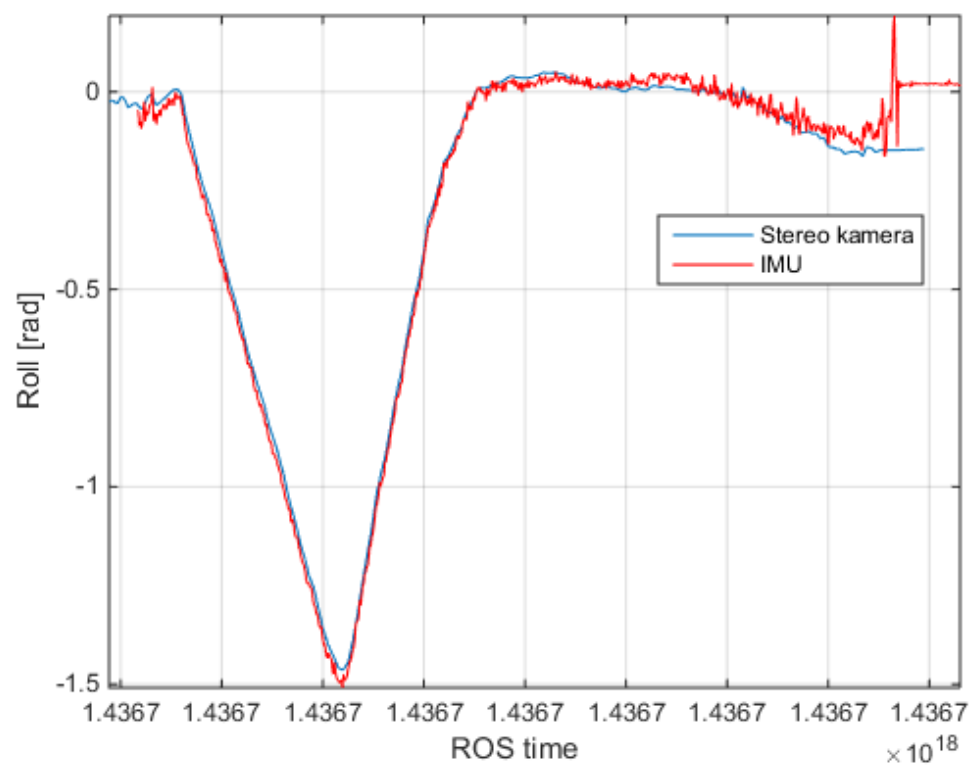
$$Roll = \arctan \frac{X}{\sqrt{Y^2 + Z^2}} \quad (4.3)$$

$$Pitch = \arctan \frac{Y}{\sqrt{X^2 + Z^2}} \quad (4.4)$$

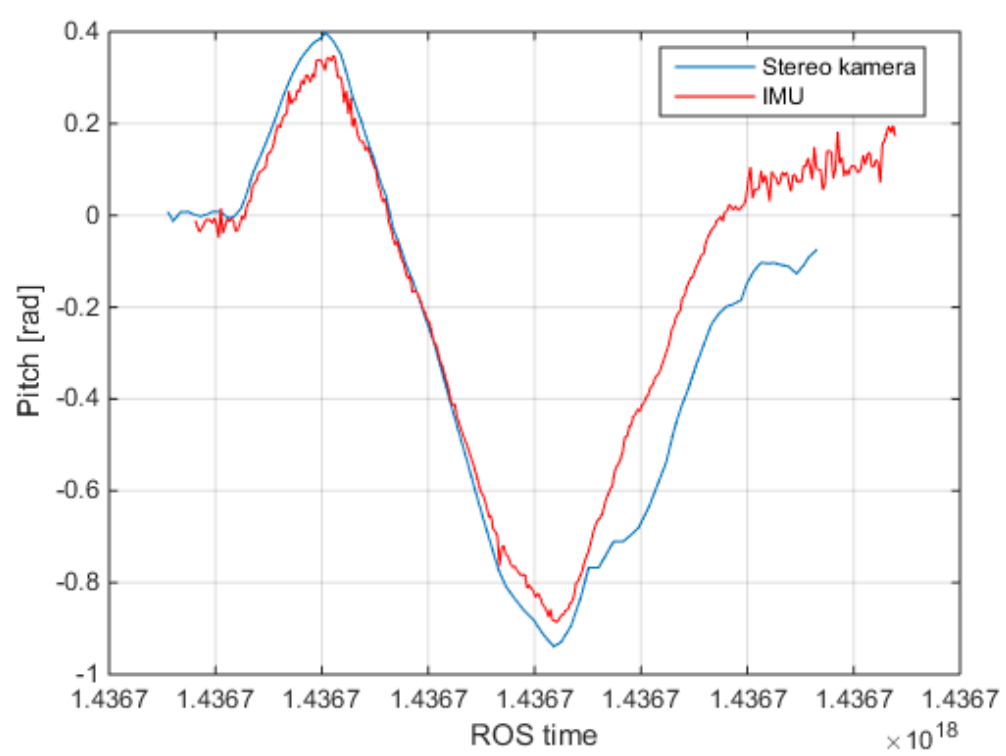
Rezultati usporedbe mjerenja prikazani su na slikama 4.6 i 4.7.

Na slici 4.6 vidimo da se mjereni kutevi potpuno podudaraju. Iz IMU-a dobivamo zašumljeniji signal, te možemo zaključiti da je stereo kamera dala preciznije rezultate. Iz slike 4.7 mjerenja se ne podudaraju u potpunosti. Jedan od mogućih razloga tome je što stereo kamera nema dovoljno detalja u slici pri procesu izdvajanja značajki kad se sustav zakrene do nekog kuta gdje se prikazuje samo strop. IMU nam je i u ovom slučaju dao više zašumljene rezultate od stereo kamere.

Rezultati lokalizacije kamere mogli bi biti i precizniji ukoliko bi *frame rate* bio veći. U eksperimentu *frame rate* kamere je 8.5 Hz (maksimalni mogući za kameru Bumblebee XB3 ako se koriste sve tri monokromatske kamere), dok je standardni iznos između 25 - 30 Hz.



**Slika 4.6:** Rezultati *Roll* kuta



**Slika 4.7:** Rezultati *Pitch* kuta

## 5. Zaključak

Kako je konvencionalna metoda komunikacije ispod vode nemoguća iz razloga jer voda zbog svojih fizikalnih svojstava iznimno jako prigušuje elektromagnetske signale, onemogućeni su i GPS sustavi te se javlja potreba za alternativnim rješenjima lokalizacije u podvodnom svijetu. Jedno od rješenja je lokalizacija korištenjem stereo kamere. Postupak lokalizacije naziva se stereo vizualna odometrija te provodi estimaciju pozicije stereo kamere u 3D prostoru.

Prije provođenja estimacije pokreta, potrebno je umjeriti odnosno kalibrirati sustav stereo kamera. Pri kalibraciji određuju se intrinzični i ekstrinzični parametri stereo sustava. Sljedeći korak je rektifikacija slike, čime se smanjuje složenost pronalaska korespondentnih značajki u lijevoj i desnoj slici. Nakon uparivanja značajki, metodom triangulacije određuje se pozicija svake značajke u 3D prostoru. Iz podataka korespondencije i triangulacije značajki, kroz dva ili više parova vremenskih okvira minimizacijom reprojekcijske funkcije pogreške određuje se vizualna odometrija. U sklopu rada korišten je postupak iz biblioteke *Libviso2*. Algoritam je testiran usporedno s IMU sustavom te se iz rezultata može vidjeti da stereo kamera daje bolje rezultate u određivanju rotacije od IMU-a, čiji su rezultati sadrže šum. Rezultati kretanja stereo sustava prema naprijed ili napredovanje prikazuju mala odstupanja od stvarnog pomaka. Bolji rezultati u određivanju translacije i rotacije stereo kamere u prostoru mogući su ako sustav kamera ima mogućnost većeg *frame rate-a*.

Scenariji da se stereo kamera prethodno našla na poziciji na kojoj se trenutno nalazi možemo provjeriti korištenjem HALOC algoritma. Algoritam koristi *hash* funkciju nad svakim video okvirom kako bi ga konvertirao u jedinstveni opis na temelju detektiranih značajki iz video okvira. Reducirani oblik video okvira se dalje koristi kako bi se uspoređivao sa svim prethodnim reduciranim video okvirima u svrhu nalaženja podudarnosti između određenih značajki kako bi se odredilo zatvaranje petlje.

U budućnosti je plan sastaviti kombinaciju algoritma za određivanje zatvaranja petlje i algoritma za određivanje vizualne odometrije u svrhu ispravljanja eventualnih pogrešaka pri estimaciji pozicije stereo kamere nakon što je prepoznato zatvaranje petlje.

Ukoliko kamera ima odstupanja u estimaciji položaja nakon što je prepoznata i uspoređena trenutna i neka prethodna pozicija korištenjem HALOC algoritma, potrebno je ispraviti prijedenu trajektoriju dobivenu vizualnom odometrijom.



# LITERATURA

- [1] Adrien Angeli, David Filliat, Stéphane Doncieux, i Jean-Arcady Meyer. Fast and incremental method for loop-closure detection using bags of visual words. *Robotics, IEEE Transactions on*, 24(5):1027–1037, 2008.
- [2] K Somani Arun, Thomas S Huang, i Steven D Blostein. Least-squares fitting of two 3-d point sets. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (5):698–700, 1987.
- [3] Herbert Bay, Tinne Tuytelaars, i Luc Van Gool. Surf: Speeded up robust features. U *Computer vision–ECCV 2006*, stranice 404–417. Springer, 2006.
- [4] Herbert Bay, Andreas Ess, Tinne Tuytelaars, i Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.
- [5] Stan Birchfield i Carlo Tomasi. Depth discontinuities by pixel-to-pixel stereo. *International Journal of Computer Vision*, 35(3):269–293, 1999.
- [6] Jason Campbell, Rahul Sukthankar, Illah Nourbakhsh, i Aroon Pahwa. A robust visual odometry and precipice detection system using consumer-grade monocular vision. U *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, stranice 3421–3427. IEEE, 2005.
- [7] Shimon Edelman, Nathan Intrator, i Tomaso Poggio. Complex cells and object recognition. 1997.
- [8] Martin A Fischler i Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.
- [9] Andrea Fusiello, Emanuele Trucco, i Alessandro Verri. A compact algorithm

- for rectification of stereo pairs. *Machine Vision and Applications*, 12(1):16–22, 2000.
- [10] Richard Hartley i Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [11] Bernd Kitt, Andreas Geiger, i Henning Lategahn. Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. U *Intelligent Vehicles Symposium (IV)*, 2010.
- [12] Ivan Krešo. Napredno estimiranje strukture i gibanja kalibriranim parom kamera. 2013.
- [13] Shu Lin, M Tamer Ozsu, Vincent Oria, i Raymond Ng. An extendible hash for multi-precision similarity querying of image databases. U *Vldb*, svezak 1, stranice 221–230, 2001.
- [14] H Christopher Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, MA Fischler and O. Firschein, eds, stranice 61–62, 1987.
- [15] David G Lowe. Object recognition from local scale-invariant features. U *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, svezak 2, stranice 1150–1157. Ieee, 1999.
- [16] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.
- [17] Vishal Monga i Brian L Evans. Perceptual image hashing via feature points: performance evaluation and tradeoffs. *Image Processing, IEEE Transactions on*, 15(11):3452–3465, 2006.
- [18] David Nistér, Oleg Naroditsky, i James Bergen. Visual odometry. U *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, svezak 1, stranice I–652. IEEE, 2004.
- [19] S Roy, X Zhu, J Yuan, i E-C Chang. On preserving robustness-false alarm tradeoff in media hashing. U *Electronic Imaging 2007*, stranice 65081C–65081C. International Society for Optics and Photonics, 2007.

- [20] Cordelia Schmid i Roger Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–534, 1997.
- [21] Hossein Shahbazi i Hong Zhang. Application of locality sensitive hashing to realtime loop closure detection. U *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, stranice 1228–1233. IEEE, 2011.
- [22] Ramarathnam Venkatesan, S-M Koon, Mariusz H Jakubowski, i Pierre Moulin. Robust image hashing. U *Image Processing, 2000. Proceedings. 2000 International Conference on*, svezak 3, stranice 664–666. IEEE, 2000.

## **Detekcija zatvaranja petlje u SLAM-u temeljenom na podvodnim vizualnim podacima**

### **Sažetak**

Zbog nemogućnosti korištenja preciznih tehnologija za lokalizaciju kao što su GPS, u podvodnim sustavima se u tu svrhu koriste drukčije metode lokalizacije među kojima je i SLAM (simultana lokalizacija i mapiranje). Pomoću SLAM algoritma, autonomna podvodna ronilica može odrediti svoju trenutnu poziciju i prijeđenu trajektoriju. Zbog mogućnosti pogreške pri određivanju trenutne pozicije, za ispravljanje se dodatno koristi HALOC algoritam. HALOC algoritam koristi hash funkciju kako bi efikasno pronašao zatvaranje petlje, odnosno prepoznao poziciju ronilice usporedbom s prethodno snimljenim video okvirima na raznim pozicijama.

**Ključne riječi:** SLAM, Detekcija zatvaranja petlje, Hash funkcija, Autonomna podvodna ronilica, Vizualna odometrija, Lokalizacija.

### **Title**

### **Abstract**

Because of the inability of using precise localization technologies such as GPS, in the underwater world localization is determined by other methods like SLAM (simultaneous localization and mapping). Using SLAM algorithm, autonomous underwater vehicle can determine its' current position and traveled trajectory. Because of the possibility of error in determining the position, we use HALOC algorithm to provide correction. HALOC algorithm uses a hash function to efficiently find the closing loop and recognize the position of the AUV by comparing with previously recorded video frames at different positions.

**Keywords:** SLAM, Loop-closure detection, Hash function, Autonomous underwater vehicle, Visual odometry, Localization.