Univerza v Ljubljani

Fakulteta za elektrotehniko

Miloš Antić

Izvedba sistema za preprečevanje trkov mobilnega sistema na podlagi detekcije ovir s stereo kamero

Magistrsko delo

Magistrski študijski program druge stopnje Elektrotehnika

Mentor: prof. dr. Igor Škrjanc

Ljubljana, 2019

Zahvala

Rad bi se zahvalil mentorju, prof. dr. Igorju Škrjancu, ki me je vodil in spremljal pri izdelavi magistrske naloge, še posebej as. dr. Andreju Zdešarju, ki mi je neizmerno pomagal, svetoval in usmerjal pri doseganju zastavljenih ciljev. Za podporo se zahvaljujem tudi celotnemu kolektivu laboratorija za kibernetiko in avtomatiko. Zahvala gre tudi mojim prijateljem, ki so me podpirali tekom študija.

Posebna zahvala gre tudi mojim staršem, sestrama in nečakinji Teodori, ki so mi omogočili brezskrben študij in me vseskozi podpirali pri premagovanju ovir.

Vsebina

Seznam slik

Seznam tabel

Seznam uporabljenih simbolov

V pričujočem zaključnem delu so uporabljene naslednje veličine in simboli:

Veličina / oznaka		**Enota**	
Ime	Simbol	Ime	Simbol
Hitrost valovanja	c	meter na sekundo	m/s
Čas	t	sekunda	s
Razdalja	d_r	meter	m
Razdalja pogleda	R_p	meter	m
Hitrost svetlobe	c_0	meter na sekundo	m/s
Topografska razdalja	$R_{topog.}$	meter	m
Kot	ε	stopinja	°
Višina	H	meter	m
Vektor	$\boldsymbol{v}_1$	-	-
Vektor	$\boldsymbol{v}_2$	-	-
Kot med vektorjema	θ_v	stopinja	°
Vektor točke na sliki	$\boldsymbol{p}_1$	-	-
Vektor točke na sliki	$\boldsymbol{p}_2$	-	-
Vektor točke v koordinatnem sistemu X	$\boldsymbol{p}_X$	-	-
Koordinata x v koordinatnem sistemu X	x_X	meter	m
Koordinata y v koordinatnem sistemu X	y_X	meter	m
Koordinata z v koordinatnem sistemu X	z_X	meter	m
Matrika notranjih parametrov	$\boldsymbol{S}$	-	-

Goriščna razdalja izražena v slikovnih elementih	f	-	-
Koordinata x središča slike	c_x	meter	m
Koordinata y središča slike	c_y	meter	m
Rotacijska matrika iz k. s. A v B	$\boldsymbol{R}_A^B$	-	-
Translacijski vektor iz k. s. A v B	$\boldsymbol{t}_A^B$	-	-
Vektor	$\boldsymbol{a}^T$	-	-
Vektor	$\boldsymbol{b}^T$	-	-
Fundamentalna matrika	$\boldsymbol{F}$	-	-
Premica na prvi sliki	$\boldsymbol{l}_{P_2}$	-	-
Premica na drugi sliki	$\boldsymbol{l}_{P_1}$	-	-
Esencialna matrika	$\boldsymbol{E}$	-	-
Razdalja med prvo in drugo kamero	b	meter	m
Dispariteta	d	-	-
Najmanjša razdalja objektov od kamere	Z_{min}	milimeter	mm
Goriščna razdalja leve kamere izražena v slikovnih elementih	f_x	-	-
Goriščna razdalja desne kamere izražena v slikovnih elementih	f_y	-	-
Intenziteta slikovnega elementa na levi sliki	I_l	-	-
Intenziteta slikovnega elementa na desni sliki	I_r	-	-
Kot nagiba	φ	stopinja	°
Kot naklona	θ	stopinja	°
Kot odklona	ψ	stopinja	°
Rotacijska matrika	$\boldsymbol{R}_x$	-	-
Rotacijska matrika	$\boldsymbol{R}_z$	-	-
Vektor pospeška	$\underline{\boldsymbol{a}}$	-	-
Matrika oblaka točk	$\boldsymbol{D}$	-	-
Rotacijska matrika	$\boldsymbol{R}$	-	-
Translacijska matrika	$\boldsymbol{T}$	-	-
Vektor stanj	$\boldsymbol{q}$	-	-

Krožna hitrost robota	ω	radian na sekundo	rad/s
Trenutni radij trajektorije robota	$R(t)$	meter	m
Razdalja med kolesi robota	L	meter	m
Radij kolesa	r	meter	m
Linearna hitrost robota	v	meter na sekundo	m/s
Linearna hitrost levega kolesa	v_L	meter na sekundo	m/s
Linearna hitrost desnega kolesa	v_R	meter na sekundo	m/s
Krožna hitrost levega kolesa	ω_L	radian na sekundo	rad/s
Krožna hitrost desnega kolesa	ω_R	radian na sekundo	rad/s
Matrika točk slike	$\boldsymbol{T}_{ovira}$	-	-
Dolžina daljice	d_r	meter	m
Radij kroga	r_o	meter	m
Koordinata x točke trka	x_c	meter	m
Koordinata y točke trka	y_c	meter	m
Radij	r_a	meter	m
Radij	r_b	meter	m
Kot med oviro in robotom	ϕ_c	stopinja	°
Čas do trka	t_{TTC}	sekunda	s
Radij trajektorije robota	R	meter	m
Pospešek	a_r	meter na kvadratno sekundo	m/s^2

Pri čemer so vektorji in matrike zapisani s poudarjeno pisavo. Natančnejši pomen simbolov ter njihovih indeksov je razviden iz ustreznih slik ali pa je pojasnjen v spremljajočem besedilu, kjer je simbol uporabljen.

Povzetek

V magistrskem delu smo raziskali problem zaznavanja okolja s stereo kamero, ki se na področju mobilnih sistemov običajno uporablja za tridimenzionalno zaznavanje okolja v obliki oblaka točk. Problem rekonstrukcije običajno rešujemo s stereo vidom, ki sloni na epipolarni geometriji. Tridimenzionalno informacijo uporabimo za gradnjo lokalnega zemljevida robota, na podlagi katerega smo implementirali algoritme za preprečevanje trkov.

Z globinsko kamero D435i pridobivamo globinsko sliko, na podlagi katere smo s triangulacijo rekonstruirali tridimenzionalno predstavitev okolja. S segmentacijo okolja smo zgradili lokalni zemljevid robota, kjer smo z morfološko operacijo dilatacije in filtrom lukenj izboljšali predstavitev dobljenega zemljevida robota. Na zemljevidu smo s segmentacijo dosegli delitev prostora na tla, ovire in nezasedene predele, tj. območja, kjer nismo uspeli pridobiti informacije iz trenutnega prizora. Na podlagi te informacije smo izvedli sistem za preprečevanje trkov pri vožnji naravnost in pri zavijanju.

Preprečevanje trkov je zasnovano na proporcionalnem zmanjšanju hitrosti glede na razdaljo do najbližje ovire oz. najmanjšem času do trka. V situacijah, ki lahko privedejo do trka z oviro, sistem za vodenje posreduje in po potrebi privede mobilni sistem v stanje mirovanja.

V magistrskem delu smo detekcijo ovir s stereo kamero in delovanje algoritmov vodenja preizkusili na robotu Pioneer 3-AT, ki se običajno uporablja za raziskovalne namene. Sistem za detekcijo ovir in preprečevanje trkov izpolnjujeta zahteve za delovanje v realnem času. Implementirani algoritmi so izvedeni v programskem orodju Matlab, ki preko brezžične povezave komunicirajo z robotom, na katerem teče meta-operacijski sistem ROS.

V tem delu smo predstavili kalibracijo sistema za detekcijo ovir, ki omogoča, da robot avtomatsko določi ravnino tal in lego stereo kamere v ravnini. Tovrstna kalibracija mobilnega sistema je pomembna tudi zaradi poenostavitve dela s sistemom (npr. v industriji), saj lahko z možnostjo avtomatičnega umerjanja odstranimo potrebo po ekspertnem znanju in zmanjšamo strošek vgradnje.

Ključne besede: avtonomni mobilni sistemi, strojni vid, stereo kamera, Pioneer 3-AT, Intel RealSense D435i, segmentacija okolja, preprečevanje trkov, detekcija ovir, merjenje globine, senzorji za zaznavanje ovir

Abstract

In the thesis we investigated the problem of environmental sensing with a stereo camera, which is commonly used in the field of mobile systems for three-dimensional sensing of the environment in the form of a point cloud. The problem of reconstruction is usually solved by stereo vision based on epipolar geometry. We use three-dimensional information to build a local robot map based on which we implemented collision avoidance algorithms.

With the D435i depth camera, we obtain a depth image, which was used for reconstruction of a three-dimensional representation of the environment through triangulation. With the segmentation of the environment we built a local map of the robot, where we improved the representation of the obtained map of the robot by means of a morphological dilation operation and a hole filter. On the map, segmentation achieved the division of space into floors, obstacles and unoccupied areas, i.e. areas where we were unable to retrieve information from the current scene. Based on this information, we implemented a system for preventing collisions when driving straight and when cornering.

Collision prevention is based on a proportional reduction of speed with respect to the distance to the closest obstacle or minimum time to impact. In situations that may result in a collision with an obstacle, the control system shall intervene and, if necessary, bring the mobile system to a standstill.

In the master's thesis we tested obstacle detection with a stereo camera and the operation of control algorithms on a Pioneer 3-AT robot, which is commonly used for research purposes. Obstacle detection and collision avoidance systems meet the requirements for real-time operation. The algorithms are implemented in the Matlab software tool, which communicate with robot via a wireless connection. A meta-operating system ROS is running on the robot.

In this thesis, we introduced the calibration of an obstacle detection system that allows the robot to determine automatically the ground plane and pose of the stereo camera in the plane. This kind of calibration is also important in order to simplify the work with the mobile system (e.g. in the industry), since the possibility of automatic calibration can eliminate the need for expert knowledge and reduce the cost of system integration.

Key words: autonomous mobile systems, machine vision, stereo camera, Pioneer 3-AT, Intel RealSense D435i, environment segmentation, collision prevention, obstacle detection, depth measurement, proximity sensors

1 Uvod

Področje avtonomnih mobilnih sistemov vzbuja zanimanje in povpraševanje v različnih industrijskih panogah in v raziskovalnih vodah. Mobilni sistemi so doživeli znaten napredek in so prisotni v vsakdanjem življenju. Razvoj mobilnih robotov oz. sistemov je pomemben za industrijo, saj želimo znižati stroške proizvodnje in optimizirati delo za večjo učinkovitost in uspešnost podjetja. Poleg minimizacije stroškov lahko z uvedbo mobilnih sistemov ali avtomatizacije povečamo produktivnost in izboljšamo kvaliteto izdelkov. Prednost mobilnih sistemov je, da lahko prevzamejo težka dela in nadomestijo prisotnost ljudi v nevarnih okoljih. Mobilni sistemi so prisotni in delujejo tudi v domačem okolju, kjer lahko opravljajo hišna opravila.

Robot ali kateri drug mobilni sistem lahko deluje avtonomno in sprejema odločitve le na podlagi zavedanja o okolju. To je mogoče le z uporabo senzorjev, ki sistemu omogočajo vpogled v okolje v katerem se nahajajo. V ta namen je v tem delu predstavljen pregled senzorjev, ki vsebuje opis načina delovanja, prednosti in slabosti vključitve tovrstnih merilnih elementov v mobilni sistem. Za lažjo predstavitev uporabe senzorjev v mobilnih sistemih je predstavljena integracija senzorjev na realnih primerih. V praksi se najpogosteje pojavlja fuzija senzorjev, kjer lahko slabosti enega senzorja odpravimo z uporabo drugega in pri tem ohranimo ustrezno informacijo, ki je potrebna za delovanje mobilnega sistema.

Da robotu omogočimo zmožnost sprejemanja odločitev pri izvrševanju zadanih nalog, je potrebno izbrati ustrezno strategijo vodenja oz. obnašanja robota. To lahko storimo tako, da združimo informacije o okolju iz vgrajenih senzorjev z izbrano strategijo vodenja in robotu omogočimo načrtovanje poti brez posega operaterja ali nadzornika. Če z razvojem algoritmov vodenja mobilnega sistema uspemo priti na določeno stopnjo avtonomnosti, lahko le-ta deluje brez prisotnosti nadzornika. Različne stopnje avtonomnosti srečamo tudi v realnem življenju (npr. asistenčni sistemi v avtomobilih).

Cilj magistrske naloge je omogočiti mobilnemu robotu zaznavanje okolja s stereo kamero in prepoznavanje ovir ter implementacijo algoritmov za preprečevanje trkov pri vožnji naravnost in pri zavijanju. V teoretičnem delu je poleg opisa senzorjev, ki se uporabljajo v mobilnih sistemih, predstavljeno tudi delovanje globinske kamere D435i proizvajalca Intel, ki smo jo uporabili v tem magistrskem delu. V praktičnem delu so prikazani načini oz. algoritmi

za obdelavo podatkov zajetih z globinsko kamero in predstavljene metode za preprečevanje trkov ter izogibanje oviram.

Struktura magistrske naloge sestoji iz štirih poglavij. V prvem poglavju je uvod, v katerem pojasnimo tematiko magistrskega dela. V drugem poglavju je predstavljen pregled senzorjev, ki se uporabljajo v mobilnih sistemih. Opisane so lastnosti, prednosti, slabosti in omejitve tovrstnih senzorjev. Temu sledi pregled sistemov za izogibanje oviram, kjer srečujemo uporabo tovrstnih senzorjev. V tretjem poglavju je predstavljeno delovanje globinske kamere D435i proizvajalca Intel, podane so njene lastnosti in teoretične osnove stereo vida, na katerem sloni stereo kamera. Temu sledi predstavitev problema zaznavanja ovir z globinsko kamero in segmentacija okolja za namen gradnje lokalnega zemljevida mobilnega sistema, na podlagi katerega lahko mobilni sistem sprejema odločitve. Četrto poglavje opisuje metode preprečevanja trkov, ki smo jih implementirali in rezultate delovanja. V zadnjem poglavju se nahaja zaključna misel, ki povzema opisane metode in rezultate magistrskega dela.

2 Zaznavanje ovir in izogibanje oviram

V tem poglavju so predstavljeni senzorji, ki jih uporabljamo pri zaznavanju ovir v mobilnih sistemih (poglavje 2.1). V poglavju 2.2 so opisani primeri uporabe stereo kamere v različnih okoljih. V poglavju 2.3 so predstavljeni realni primeri vključitve opisanih senzorjev in metod vodenja.

2.1 Senzorji za zaznavanje ovir

Predstavljene so lastnosti senzorjev, ki jih uporabljamo pri izgradnji avtonomnih mobilnih sistemov. Osredotočili smo se na prednosti in slabosti posameznih senzorjev kot tudi na omejitve, ki skupaj tvorijo okvire uporabe merilnih elementov v tovrstnih sistemih.

Da bi mobilni sistem lahko avtonomno deloval v določenem okolju, se ga mora najprej zavedati. Vpogled v okolje omogočajo senzorji, ki delujejo na podoben princip kot čutila pri ljudeh. Če jih ne bi imeli, se ne bi mogli izogibati oviram ali pa načrtovati poti, saj ne bi imeli nikakršne predstave o okolju. Poleg pridobivanja informacij iz okolja, lahko s senzorji pridobivamo informacije o notranjih stanjih sistema, tj. mobilnega robota.

Senzorje lahko razčlenimo na aktivne in pasivne senzorje. Aktivni so tisti, ki oddajajo v okolje neko energijo oziroma valovanje preko katerega izvajajo meritve. Primera aktivnih senzorjev sta radar in ultrazvočni merilnik razdalje. Na drugi strani imamo pasivne senzorje, ki za izvedbo meritev uporabljajo energijo iz okolja. Primera pasivnih senzorjev sta navadna kamera brez osvetlitve in kompas.

Pomembne karakteristike senzorjev so [1]:

- **Območje** definira zgornjo in spodnjo mejo merjene količine, ki običajno nista simetrični. Senzorje je priporočljivo uporabljati v predpisanem območju, vsakršna prekoračitev območja lahko rezultira v poškodbi senzorja.
- **Resolucija** je najmanjša sprememba merjene količine, ki jo senzor lahko zazna. Če ima senzor analogno-digitalni (AD) pretvornik, potem je resolucija senzorja enaka resoluciji pretvornika (npr. resolucija senzorja z 10-bitnim AD-pretvornikom in območjem 5 V je $\frac{5V}{2^{10}}$).

- **Občutljivost** je sprememba izhodne vrednosti senzorja na enoto merjene količine (npr. merilnik razdalje meri razdaljo, ki je dana kot napetost na izhodu senzorja). Občutljivost je lahko konstantna čez celotno območje (linearnost) ali pa se spreminja (nelinearnost).
- **Linearnost** senzorja je njegova lastnost, da je izhod senzorja linearno odvisen od merjene količine čez celotno območje. Linearen senzor ima konstantno občutljivost v celotnem območju.
- **Pasovna širina** oz. največja hitrost podajanja informacije se nanaša na frekvenco s katero lahko senzor zajema oz. podaja meritve in je podana v Hz. Pasovna širina je največja frekvenca ali hitrost vzorčenja, pri kateri je izmerjenih samo 70,7 % pravih vrednosti.
- **Natančnost** je določena s pričakovano napako merjenja, ki je razlika med izmerjeno (m) in pravo vrednostjo (v). Natančnost se izračuna iz relativne napake merjenja kot $natančnost = 1 - \frac{|m-v|}{v}$.
- **Ponovljivost** je stopnja ponovljivosti meritev senzorja pri isti pravi vrednosti merjene količine. Dejanski izhod senzorja proizvede obseg vrednosti, medtem ko je prava vrednost izmerjena večkrat. Ponovljivost je povezana z varianco meritev.
- **Sistematična** ali **deterministična napaka** je posledica dejavnikov, ki jih lahko predvidimo ali modeliramo (pristranskost, temperaturno lezenje, kalibracija senzorja, popačenje, ki je posledica leče kamere itn.).
- **Naključna** ali **nedeterministična napaka** je nepredvidljiva in jo lahko opišemo le s funkcijo gostote verjetnosti (npr. normalna porazdelitev). To napako pogosto imenujemo šum in je običajno značilna za razmerje signal-šum.

V tabeli 1 so predstavljeni senzorji, ki jih uporabljamo za zaznavanje ovir [1].

Tabela 1. Klasifikacija senzorjev v mobilni robotiki glede na njihovo aplikacijo ali namen (meri notranja stanja [N]/meri zunanja stanja [P]) in oddano energijo (aktivni senzor [A]/pasivni senzor [P])

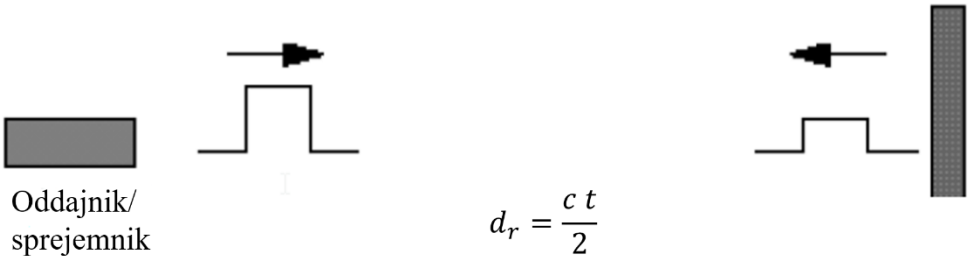
Razvrstitev	**Uporabnost**	**Senzorji**	**N/Z**	**A/P**
Taktilni in haptični senzorji	Zaznavanje trčenja, varnostna zaustavitev, bližina	Kontaktna stikala	Z	P
		Senzor bližine	Z	P
		Niz senzorjev tlaka	Z	P
Senzorji razdalje	Merjenje razdalje do ovir, lokalizacija, čas potovanja (angl. *time-of-flight*)	Ultrazvočni senzorji	Z	A
		Laserski pregledovalniki razdalj		A
		Kamera		A/P
Senzorji, ki temeljijo na strojnem vidu	Identifikacija, razpoznavanje objektov, sledenje objektu, lokalizacija, segmentacija	Kamera	Z	P/A
		Globinska kamera	Z	A
		Stereo kamera	Z	P/A
		RFID	Z	A
		Radar	Z	A
		Optična triangulacija	Z	A

Ultrazvočni senzorji, laserski pregledovalniki razdalj, globinske kamere, radarji in infrardeči senzorji v osnovi temeljijo na principu merjenja časa preleta (uveljavljena je kratica ToF – angl. *Time Of Flight*) [1]. Izhod teh senzorjev je meritev razdalje, ki jo ocenimo preko meritve časa potovanja valovanja (zvoka ali svetlobe). Na sliki 1 je prikazan princip merjenja razdalje, kjer merimo čas tako, da senzor odda valovanje in čaka na prejeto odbito valovanje. Nato iz časa potovanja valovanja pri znani hitrosti (hitrost svetlobe $c = 3 \cdot 10^8 \frac{m}{s}$, hitrost zvoka $c = 340 \frac{m}{s}$) ocenimo razdaljo do predmeta ali ovire z enačbo

$$d_r = \frac{ct}{2}, \tag{1}$$

kjer je c hitrost svetlobe in t čas potovanja valovanja. V enačbi (1) se v imenovalcu nahaja faktor dve, in sicer zato, ker nas zanima čas, ki smo ga potrebovali da smo zaznali oviro oz. ko se je valovanje odbilo [1]. Na ta način lahko izračunamo razdaljo do ovire,

ne pa celotne poti, ki jo je valovanje opravilo. Natančnost meritve razdalje je odvisna od natančnosti merjenega časa, kar je pri velikih hitrosti valovanja problematično. V tem primeru lahko točnost meritve izboljšamo z izračunom povprečja več meritev. Čas potovanja se lahko meri s korelacijskimi metodami ali z uporabo interferometrije.



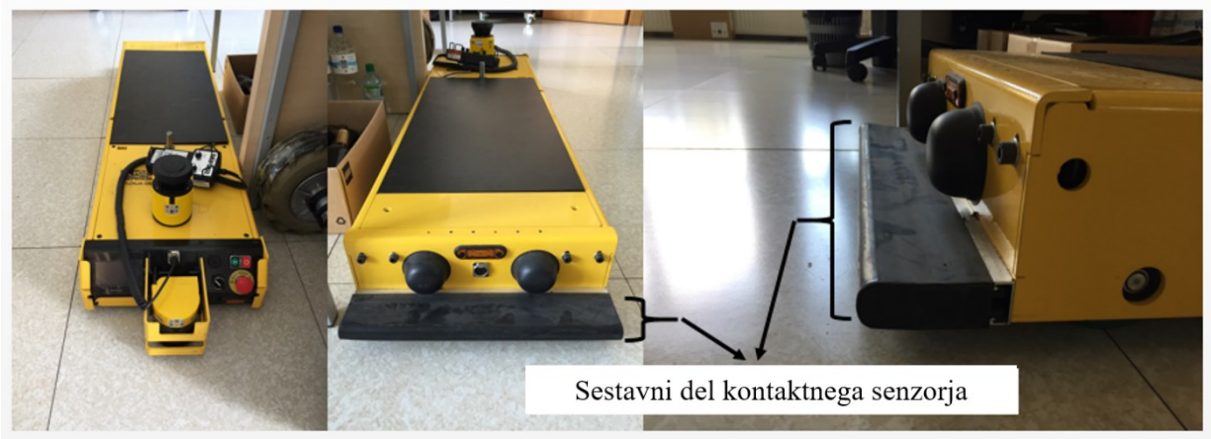
Slika 1. Princip meritve razdalje na osnovi merjenja časa potovanja valovanja.

Primeri tovrstnih senzorjev so:

- ultrazvočni senzorji razdalj,
- laserski pregledovalniki razdalj,
- radar in
- globinska kamera (ToF-kamera)

2.1.1 Taktilni senzorji

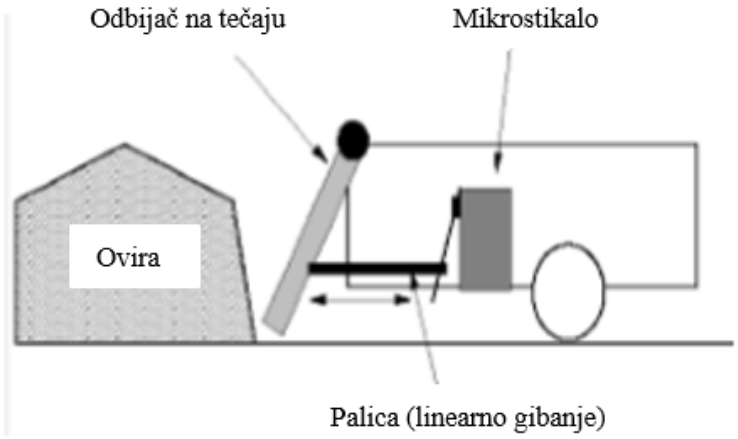
Primer taktilnih senzorjev so zelo enostavni kontaktni senzorji, ki so zelo pogosti pri enostavnih mobilnih sistemih za detekcijo ovir [2], kot je robotski sesalnik Roomba, ki je podrobneje opisan v podpoglavju 2.3.1. Ostali tipični predstavniki so optične bariere, merilniki bližine in podobno. Poleg mobilnih sistemov, ki so prilagojeni za domače okolje, poznamo tudi tiste, ki delujejo v industrijskem okolju. Na sliki 2 je prikazan primer avtomatsko vodenega vozila skupine TPV [3], ki ima na zadnjem delu vgrajen kontaktni senzor, ki se sklene, ko mobilni sistem naleti na oviro pri vzvratni vožnji.



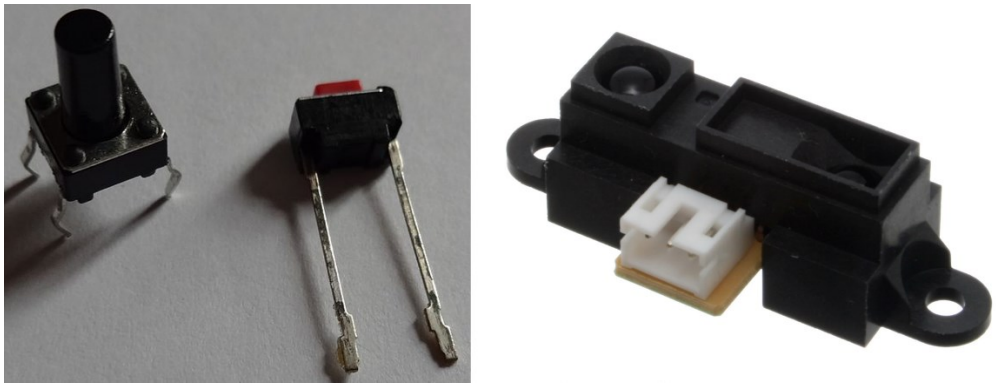
Slika 2. Avtomatsko vodeno vozilo

V primeru kontaktnih senzorjev gre v osnovi za kontakt, ki se ob naletu na oviro sklene. Na sliki 3 je predstavljeno osnovno načelo delovanja senzorja na dotik. Ko se robot dotakne ovire, se preko odbijača palica linearno premakne in sklene stikalo ter jo tako zaznamo. Taktilni senzorji so skoraj vedno prisotni na mobilnih sistemih. Če so aktivirani, se mora mobilni sistem obvezno ustaviti in prenehati z izvajanjem nalog. Npr. ko robotski sesalnik naleti na oviro oz. zid se mora ustaviti in spremeniti smer gibanja oz. sesanja.

Taktilni senzorji torej merijo informacijo preko fizične interakcije z okoljem. Splošno so modelirani po biološkem občutku kožnega dotika, ki so posledica mehanske stimulacije in temperature [4]. Primera kontaktnega stikala in infrardečega merilnika bližine sta prikazana na sliki 4.



Slika 3. Princip delovanja kontaktnega senzorja



Slika 4. Kontaktno stikalo (levo), infrardeči merilnik bližine (desno)

Leva fotografija: Vahid alpha [CC BY 3.0] (https://commons.wikimedia.org/wiki/File:Micro_switch.jpg)
Desna fotografija: oomlout [CC BY-SA 2.0] (https://www.flickr.com/photos/snazzyguy/4150703942/in/photostream/)

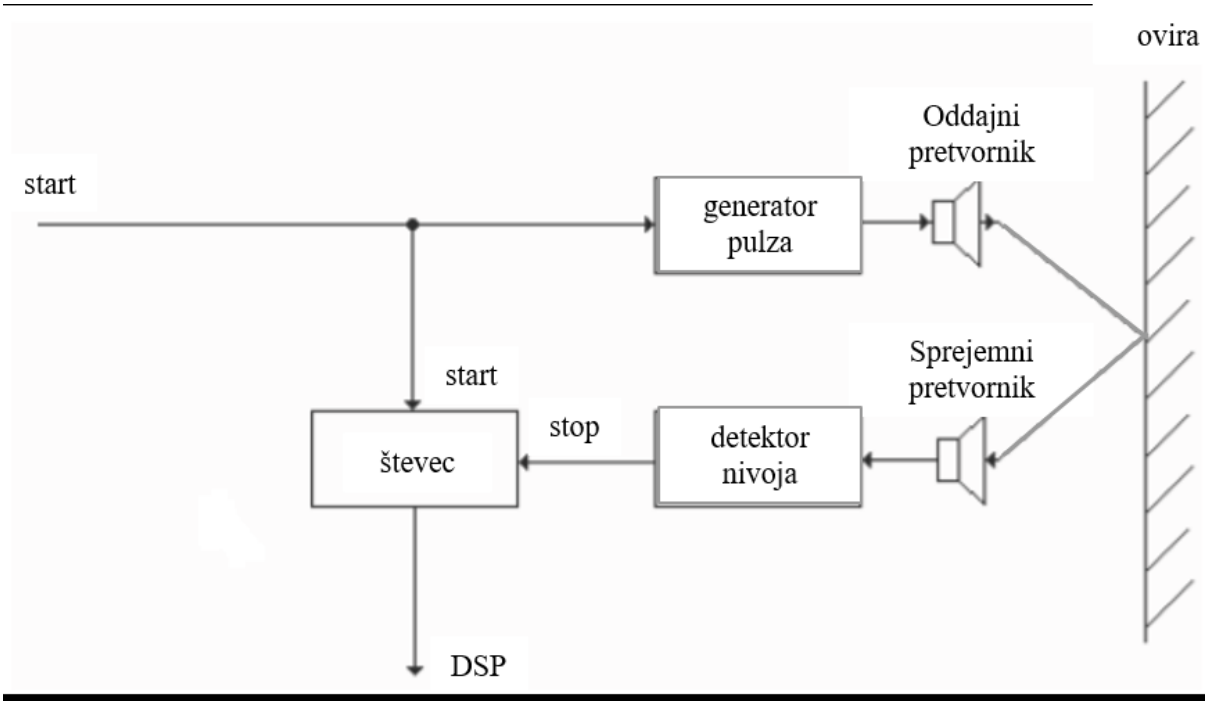
Uporabljamo jih v robotiki, računalniški strojni opremi, varnostnih sistemih, na napravah ali zaslonih na dotik, gumbih v dvigalih itn. [5]. Uporabnost taktilnih senzorjev je prisotna v testiranju delovanja aplikacij v avtomobilski industriji, kjer se testirajo zavore, sklopke ali tesnila vrat [5]. Nekatere metode za hkratno lokalizacijo in preslikavo temeljijo na

taktilnih senzorjih [6]. Imamo več izvedb senzorja, vključno z piezoelektričnim, piezouporovnim in kapacitivnim senzorjem [7].

Senzorji, ki merijo zelo majhne spremembe, morajo imeti zelo velike občutljivosti in morajo biti zasnovani tako, da imajo majhen učinek na to, kar merijo [5]. Z izdelovanjem manjših senzorjev se te karakteristike izboljšajo.

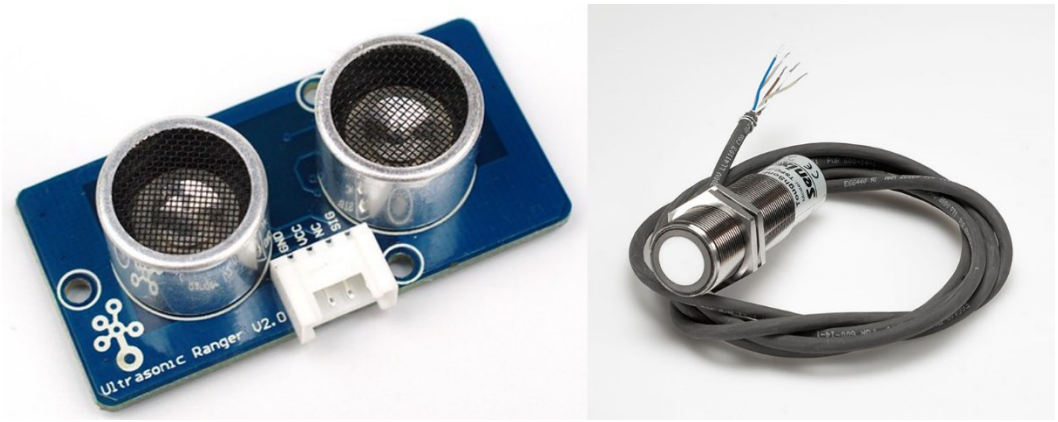
2.1.2 Ultrazvočni senzorji

Delujejo na principu odbitih valov zvoka in z njimi merimo razdalje do objektov z merjenjem časa med oddanim in prejetim valom [2]. Senzor vsebuje ultrazvočni oddajnik in sprejemnik. Na sliki 5 je prikazana osnovna zgradba senzorja, ki vsebuje na oddajni strani generator pulzov in oddajni pretvornik ter na sprejemni strani sprejemni pretvornik in detektor nivoja. S start signalom sporočimo števcu začetek štetja, ki ga končamo, ko prejme signal stop s strani detektorja nivoja. Izhodni signali števca se nato obdelajo na digitalnem signalnem procesorju (DSP), kjer se izvede natančno merjenje časa.



Slika 5. Osnovni princip zgradbe ultrazvočnega senzorja

Slika 6 prikazuje primer dveh senzorjev. Ultrazvočni senzorji nimajo velikega dometa in niso usmerjeni, ker oddajajo v snopu nekaj 10°.



Slika 6. Ultrazvočni senzor

Leva fotografija: Gareth Halfacree [CC BY-SA 2.0] (https://www.flickr.com/photos/120586634@N05/22805671435)
Desna fotografija: Bpducharme [CC BY-SA 4.0] (https://commons.wikimedia.org/wiki/File:Senix_ToughSonic_14_Ultrasonic_Sensor.jpg)

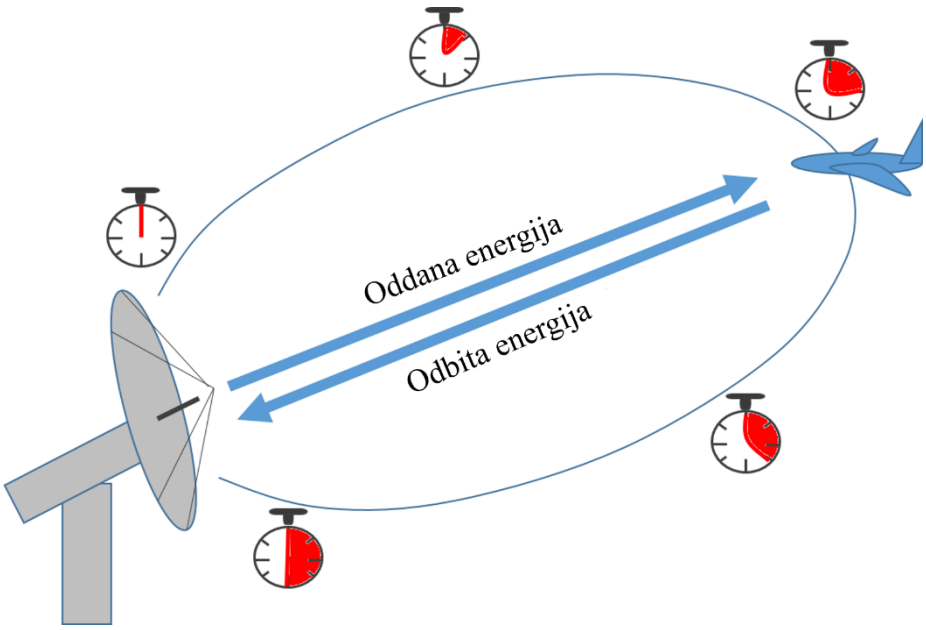
Jakost odbitega valovanja je odvisna od površine predmeta oz. zakona odbojnosti, kar pomeni določeno negotovost pri zaznavanju [2]. Ko se zvok odbije od predmeta, se ga le manjši del odbije proti senzorju, nekaj se ga razprši, nekaj pa gre skozi predmet. Možni so tudi večkratni odboji, kar vodi do napačnih meritev, ali odboji mimo senzorja. Na točnost meritve vpliva tudi temperatura okolice, saj že sprememba temperature za 17 °K povzroči napako 0,3 m na razdalji 10 m, tj. 3 % napako razdalje [2]. Poleg tega pa temperatura kot tudi medij, po katerem se širi zvok, spreminjata hitrost širjenja zvoka.

Ultrazvočni senzorji so v večini primerov neobčutljivi na svetlobo, meglo, prah in dim. Ti senzorji se pojavljajo tudi v izvedbi sonarjev, saj zvok lahko prodre vodo.

Tovrstni senzorji so prisotni v avtomobilskih aplikacijah za merjenje razdalj [8], v sistemih za izogibanje oviram z uporabo ultrazvočnega senzorja, ki meri umetno potencialno polje [9]. Če želimo uporabiti več senzorjev hkrati, moramo poskrbeti, da se med seboj ne motijo. To lahko storimo tako, da jih prožimo izmenično.

2.1.3 Radarski senzorji

Beseda radar izhaja iz kratice za radijsko detekcijo in merjenje razdalj (angl. *RAdio Detection And Ranging*). Delovanje radarskih sistemov smo povzeli po prispevku [10]. Sistem deluje tako, da antena oz. radar oddaja radijske valove oz. pulze z visoko oddajno močjo [11]. Ti impulzi so osredotočeni v eno smer, glede na usmerjenost antene in se v tej smeri širijo s hitrostjo svetlobe. Ko le-ti dosežejo trdno snov, se odbijejo nazaj proti anteni. Antena jih zazna kot odmev, jih analizira in s tem določi obliko, smer, velikost, koordinate in hitrost objekta. Moč prejetega odmeva je sorazmerna z velikostjo in obliko radarsko odbojnih površin (vsi materiali ne odbijajo radijskih valov). Radarski sistemi so aktivni, ker oddajajo energijo v okolje za izvedbo meritve. Na sliki 7 je prikazan princip radarske tehnologije, ki se uporablja v letalskih aplikacijah.



Slika 7. Princip radarja: merjenje časa potovanja oddanega in nato odbitega pulza

Razdaljo lahko merimo tudi z navadnim osciloskopom. Na osciloskopu se osvetljena točka premika sinhrono z oddanim pulzom in pušča sled. Odklon se pojavi, ko oddajnik odda pulz. Osvetljena točka se s pomočjo radijskih valov premakne na lestvici na osciloskopu. V trenutku, ko antena prejme odmev oz. odbojni pulz, je ta prikazan tudi na osciloskopu. Razdalja med dvema prikazanima impulzoma na osciloskopu je merilo razdalje.

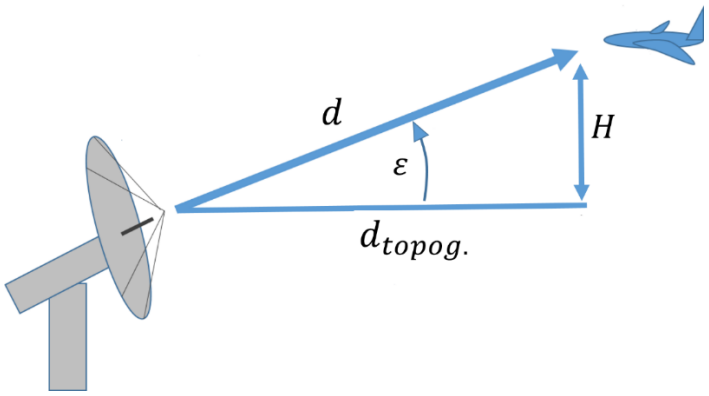
Ker se elektromagnetna energija ali radijski valovi širijo s konstantno hitrostjo, tj. svetlobno hitrostjo, lahko določimo razdalje med odbojnimi predmeti (letala, ladje ali avtomobili) in radarskim mestom z merjenjem časa potovanja oddanih impulzov. Dejanska razdalja med radarjem in objektom se imenuje razdalja pogleda (angl. *slant range*), ki jo lahko izračunamo po enačbi (1).

Ker valovi potujejo do cilja in nazaj, se čas potovanja deli s faktorjem dve, da dobimo čas, ki ga je val potreboval, da je dosegel cilj. Na sliki 8 je prikazan primer uporabe radarja za zaznavanje položaja letal. Z radarjem izmerimo razdaljo d do letala. Topografsko razdaljo $d_{topog.}$ do letala lahko izračunamo po enačbi (2), pri čemer nismo upoštevali zemljine ukrivljenosti

$$d_{topog.} = d \cos \varepsilon \,, \tag{2}$$

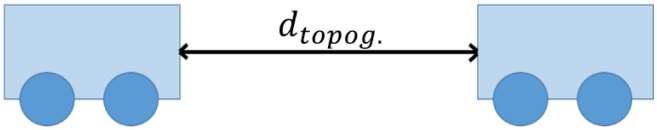
kjer je ε kot.

Višino H, na kateri se nahaja letalo, lahko uporabimo v izračunu topografske razdalje, kjer upoštevamo tudi Zemljino ukrivljenost.



Slika 8. Razdalja pogleda in topografska razdalja trigonometrične povezave brez upoštevanja Zemljine ukrivljenosti

V avtomobilskih aplikacijah običajno nimamo opravka s kotom ε, ker se vozila nahajajo približno na isti višini (slika 9).



Slika 9. Topografska razdalja v avtomobilskih aplikacijah

Če je v smeri širjenja valov ovira, na primer letalo ali avtomobil, potem je del energije pulza razpršen v vse smeri. Zelo majhen del se odbije nazaj do radarja. Radarska antena sprejema to energijo in radar ocenjuje vsebovano informacijo.

Na radarsko tehnologijo ne vplivajo neugodne vremenske razmere kot so oblaki, padavine, dež, megla in lahko prodre celo skozi stene in plasti snega. Senzorji lahko delujejo podnevi ali ponoči tudi na dolgih razdaljah. Poleg zaznavanja je možno sledenje premikajočim se objektom in prepoznavanje objekta s slikanjem z visoko ločljivostjo. Z radarskim valom lahko zaznamo več objektov hkrati, ne omogoča pa odkrivanja manjših predmetov zaradi daljše valovne dolžine, kar posledično daje popačene ali nezadostne rezultate. Prednost radarskih senzorjev je, da materiali, ki se štejejo za izolatorje, kot so guma in plastika, ne ovirajo zbiranja podatkov radarskih signalov. Signali bodo prodrli v materiale in zajeli potrebne podatke. Tako lahko senzor skrijemo za odbijač v primeru avtomobilskih aplikacij.

Če uporabljamo radarsko tehnologijo na daljših obratovalnih razdaljah, je potrebno več časa, da se pridobijo podatki o objektu. V primeru objektov, ki za radarsko tehnologijo nimajo primerne odbojnosti, jih v tem primeru ne bomo zaznali. Radijski signali imajo težave z objekti, ki se nahajajo za prevodniki. V tem primeru senzor zelo težko pridobi podatke o želenem cilju. Radijski signali potujejo skozi zrak in prostor, kjer lahko potujejo tudi drugi signali drugih frekvenc. To lahko povzroči motnje v oddanem signalu, če ga ne usmerimo

pravilno in s tem povzročimo spremembo prenesene informacije in posledično napačno meritev.

Poznamo več vrst radarskih senzorjev:

- Dopplerjev radar,
- bistatični radar,
- monopulzni radar,
- pasivni radar,
- FM-radar,
- vremenski radar,
- radar za kartiranje,
- navigacijski radar s krajšimi valovnimi dolžinami itn.

Radarski senzorji so uporabljeni v avtomobilskih aplikacijah, kot so adaptivni tempomat, zaviranje v sili, daljinsko zaznavanje razlitega olja in spremljanja zdravja z biomedicinskimi radarji [12]. Na sliki 10 je predstavljen primer industrijskega radarskega senzorja proizvajalca Symeo.

Slika 10. Industrijski radar proizvajalca Symeo

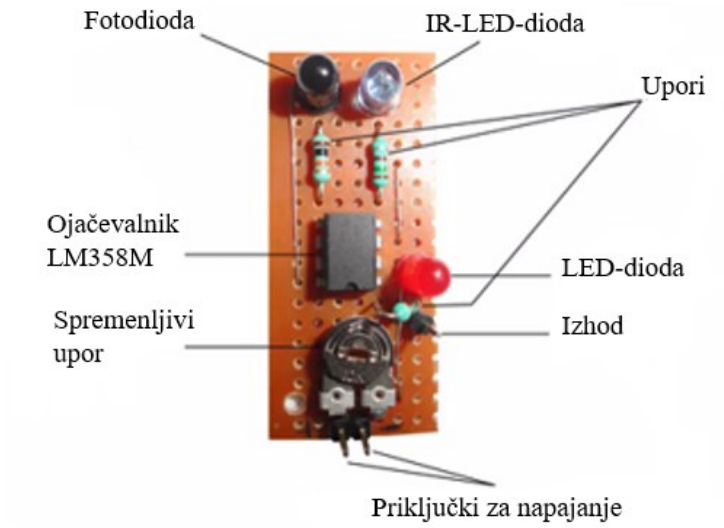
Fotografija: Hihiman [CC0] (https://de.wikipedia.org/wiki/Radar)

2.1.4 IR-senzorji

IR-senzor je naprava, ki zaznava infrardeče sevanje z valovnimi dolžinami od 700 nm vse do 1 mm [13]. Merimo lahko spremembe v toploti, temperaturi, gibanju, vlagi, zvoku itn. Tovrstni senzorji temeljijo na Planckovem, Stefan-Boltzmannovem in Weinovem zakonu. Senzor je sestavljen iz para polprevodnikov, in sicer iz infrardeče LED-diode, ki je oddajnik in iz fototranzistorja ali IR-fotodiode, ki je sprejemnik in je občutljiva na infrardečo svetlobo z enako

valovno dolžino kot tista, ki je bila izsevana. IR-dioda je običajno narejena iz galijevega arzenida ali aluminij galijevega arzenida.

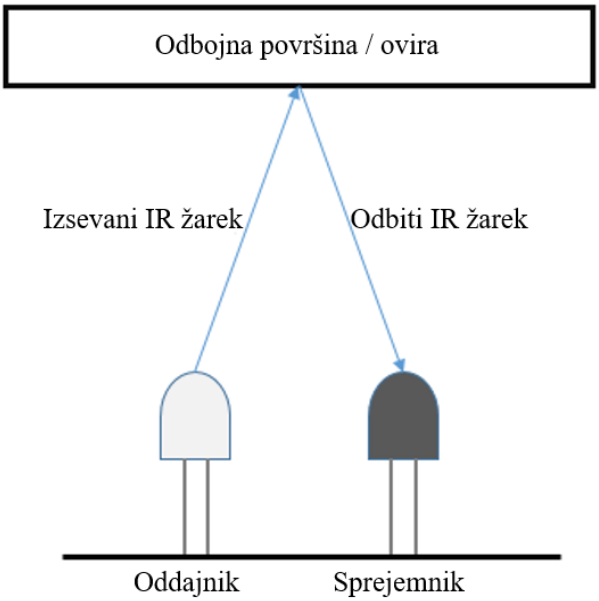
Glavna funkcija oddajnika je, da pretvori električno energijo v svetlobo. Deluje na principu rekombinacije nosilcev naboja [14]. To pomeni, ko dovedemo električno energijo, elektron preide iz valenčnega pasu v prevodni pas in tako nastane par elektron-vrzel. Ta proces se imenuje generacija. Za to, da se elektron vrne nazaj v valenčni pas je potrebno, da odda dovolj energije, pri tem pa izgine tudi vrzel v valenčnem pasu. Oddana energija se pretvori v foton, tj. delec, ki predstavlja izsevano svetlobo. Izvori infrardečega sevanja so poleg LED-diod vsa telesa, ki imajo temperaturo nad nič Kelvinov. V večini primerov so to infrardeči laserji, črna telesa, volframove žarnice itn. Upornost in izhodna napetost na fotodiodi pa se spreminja proporcionalno z velikostjo sprejete infrardeče svetlobe. Na sliki 11 je prikazano vezje infrardečega senzorja.



Slika 11. Vezje infrardečega senzorja

Fotografija: Vishwam Aggarwal [CC BY-NC-ND 3.0] (http://maxembedded.com/2013/08/how-to-build-an-ir-sensor/) Dostopano: 29.08.2019, slika je prevedena.

Zaznavanje objektov z infrardečimi senzorji se začne najprej z oddajanjem sevanja želene valovne dolžine v okolje, ki se od objekta odbije nazaj proti detektorju. Ta zazna odbito valovanje in nam glede na njegovo intenziteto vrne izhodni signal, ki je navadno majhen, zato uporabljamo ojačevalce, da zaznamo signal. Na sliki 12 je prikazano delovanje infrardečega senzorja.



Slika 12. Princip meritve infrardečega senzorja

Poznamo dve možni postavitvi sprejemnika in oddajnika [14]. V prvem primeru med njima ni ovir, kar pomeni, da sta praktično postavljena nasproti in med njima poteka nevidna linijska povezava. Če bi med njiju postavili oviro, bi ta ovirala povezavo in sevanje ne bi prišlo do sprejemnika. Senzor lahko na ta način uporabimo v protivlomnih sistemih. V drugem primeru pa sta postavljena eden poleg drugega, iskani objekt pa pred njima kot je to prikazano na sliki 12. Preko odboja valovanja od objektov zaznavamo potencialne ovire.

Do sedaj smo obravnavali aktivni infrardeči senzor, ki vsebuje oddajnik in sprejemnik. Pasivni na drugi strani vsebujejo le sprejemnik oz. detektor. Taki senzorji za izvor infrardečega sevanja uporabljajo objekte v okolici.

Pasivne senzorje uporabljamo v sistemih za detekcijo človeških teles [15] ali piroelektrilne senzorje v varnostnih sistemih [16]. Aktivni senzor uporabljamo tudi v robotskih aplikacijah za lokalizacijo v prostorih [17].

Prednosti IR-senzorja [18]:

- Zaradi zahtev po nizki porabi so primerni za večino elektronskih naprav, kot so prenosniki, telefoni in dlančniki.
- Zmožni so zaznavanja gibanja v prisotnosti ali odsotnosti svetlobe s skoraj isto zanesljivostjo in ne potrebujejo fizičnega kontakta za odkrivanje ovir.
- Zaradi usmerjenosti žarka IR-sevanja ni uhajanja podatkov. Korozija in oksidacija ne vplivata na senzor, ki ima močno odpornost proti šumu.

Slabosti IR-senzorja so [18], da zahteva vidno linijo in ima omejen obseg. Na senzor lahko vplivajo razmere v okolici, kot so dež, megla, prah in onesnaženje. Poleg tega ima počasen prenos podatkov.

2.1.5 LIDAR

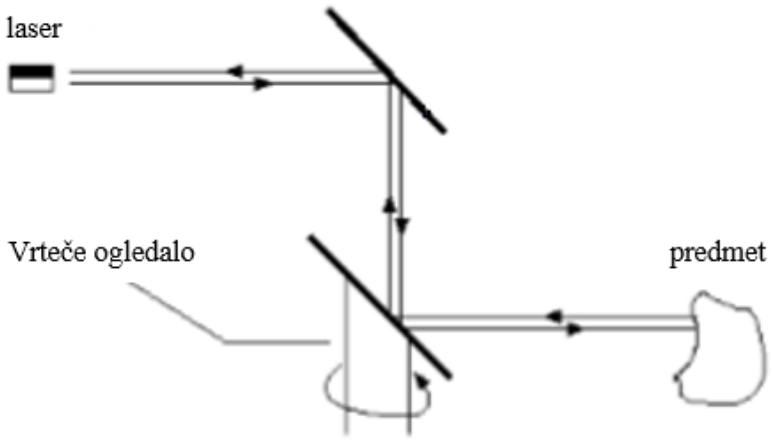
LIDAR je kratica za zaznavanje svetlobe in določevanje razdalje do cilja (angl. *Light Detection And Ranging*). V nadaljevanju bomo opisali tehnologijo LIDAR, ki smo jo povzeli po [19]. LIDAR se nanaša na tehnologijo zaznavanja na daljavo, ki oddaja intenzivne, usmerjene žarke svetlobe in meri čas, ki je potreben, da senzor zazna odboje. To informacijo uporabimo za izračun območij ali razdalj do ovir. Na ta način je LIDAR analogen radarju, vendar se razlikujeta v tem, da LIDAR sloni na diskretnih pulzih laserskega žarka. Tridimenzionalne koordinate (x, y, z ali zemljepisna širina, višina in dolžina) ciljnih objektov izračunamo iz:

- časovne razlike oddanega in prejetega laserskega žarka,
- kota, pri katerem smo žarek prožili in
- absolutne lokacije senzorja na ali nad površino zemlje.

Pridobivanju podatkov z laserskimi žarki pravimo tudi lasersko skeniranje, kjer žarkom spreminjamo smer z [2]:

- vrtečim laserjem,
- sistemom vrtečih zrcal, ki so lažja, hitreje se lahko vrtijo in so bolj natančna.

Na sliki 13 je prikazan princip izvedbe merilnika. Z vrtenjem zrcala usmerjamo žarke in s tem omogočimo, da merimo razdaljo odboja žarka pri več kotih, npr. v območju od 0° do 180° po korakih 0,5° [2]. Tako merimo razdalje do ovir v ravnini merjenja, kar je pogosta izvedba laserskega merilnika. LIDAR oz. laserski pregledovalnik razdalj omogoča od deset tisoč do sto tisoč meritev razdalj v sekundi.



Slika 13. Princip meritve razdalje z laserskim pregledovalnikom

Ločimo LIDAR sisteme, ki temeljijo na različnih platformah [20]:

- zemeljski LIDAR,
- zračni LIDAR,
- vesoljski LIDAR.

Na sliki 14 so prikazani trije tipi sistemov LIDAR, ki jih lahko uporabljamo za aplikacije v vesolju, v zraku ali na zemlji.

Slika 14. Lidar sistemi glede na platformo

Leva fotografija: David Monniaux [CC BY-SA 3.0] (https://en.wikipedia.org/wiki/Lidar)
Sredinska fotografija: Cargyrak [CC BY-SA 4.0] (https://en.wikipedia.org/wiki/Lidar)
Desna fotografija: Charly W. Karl [CC BY-ND 2.0] (https://www.flickr.com/photos/cwkarl/28261160594)

Naprej delimo LIDAR sisteme glede na fizikalne procese in vrste sipanja. Tipi glede na fizikalne procese [20]:

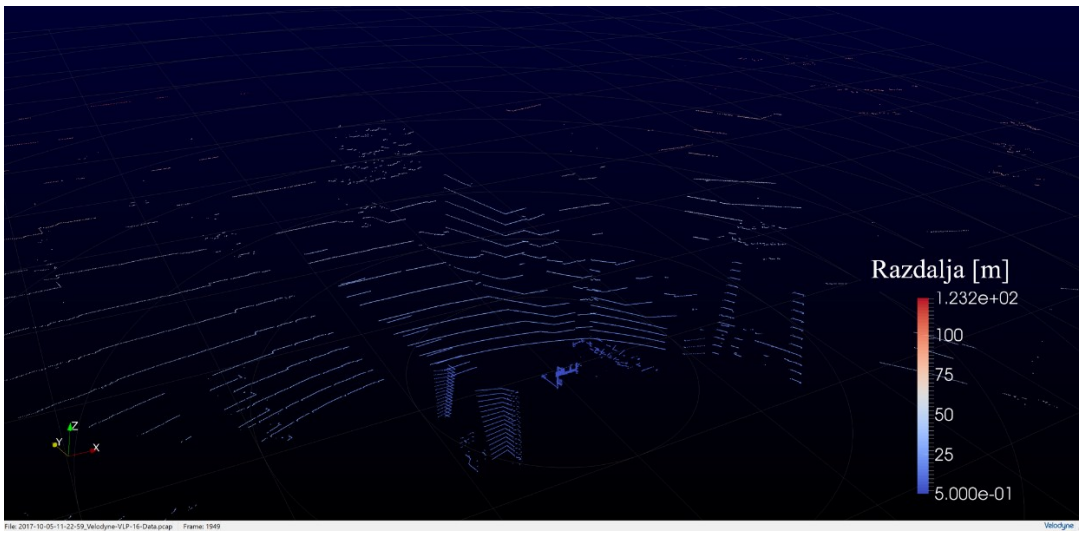
- daljinomer LIDAR,
- diferencialni absorpcijski LIDAR in
- Dopplerjev LIDAR.

Tipi glede na vrsto sipanja [20]:

- Mie,
- Rayleigh,
- Raman in
- fluorescenčni.

LIDAR-sistemi so aktivni sistemi [19], saj oddajajo impulze svetlobe v okolje in zaznavajo odboje. Ta značilnost omogoča, da se lidarski podatki zbirajo tudi ponoči, npr. v avtomobilskih, letalskih in drugih aplikacijah. Za razliko od radarja, LIDAR ne more prodreti v oblake, dež ali gosto meglico, zato moramo podatke zbirati med lepim vremenom ali ponoči. Tako lahko merimo površino Zemlje pri hitrostih vzorčenja večjih od 150 kHz, tj. 150000 impulzov na sekundo. Rezultat meritev je gosto razporejena mreža zelo natančnih georeferenciranih višinskih točk, ki se pogosto imenuje oblak točk in se lahko uporabi za

ustvarjanje tridimenzionalnih predstavitev Zemlje. Na sliki 15 je prikazan primer pridobivanja meritev razdalje s senzorjem LIDAR v avtomobilskih aplikacijah.



Slika 15. 3D globinska slika dobljena iz sistema LIDAR

Mnogi LIDAR-sistemi delujejo v skoraj infrardečem območju elektromagnetnega spektra, čeprav nekateri senzorji delujejo tudi v zelenem pasu, da prodrejo v vodo in zaznajo značilnosti dna.

Naštete lastnosti senzorja nakazujejo uporabo v številnih okoljih. Poznamo realne primere notranje in zunanje uporabe [21]:

- Avtonomna vozila lahko uporabljajo LIDAR za zaznavanje ovir in izogibanje za varno krmarjenje skozi okolje [22]. Izhodni podatki oblaka točk iz senzorja LIDAR zagotavljajo potrebne podatke za robotsko programsko opremo, da bi ugotovili, kje v okolju obstajajo potencialne ovire in kje je robot povezan s temi morebitnimi ovirami.
- Tehnologija LIDAR se v avtonomnih mobilnih sistemih uporablja za zaznavanje okolja in klasifikacijo objektov [23]. Sposobnost tovrstne tehnologije, da zagotovi tridimenzionalne višinske zemljevide terena, visoko natančnost oddaljenosti od tal in hitrost približevanja, omogoča varno pristajanje robotskih in voznih sredstev z veliko natančnostjo [24]. LIDAR se pogosto uporablja tudi za hkratno lokalizacijo in gradnjo zemljevida [25].
- Sistemi LIDAR se uporabljajo tudi za izboljšanje gospodarjenja z gozdovi [26]. Meritve se uporabljajo za popisovanje gozdnih parcel in za izračun posameznih višin dreves, širine krošnje in premera krošnje. Druge statistične analize uporabljajo izhodne podatke za oceno podatkov, kot so volumen krošnje, srednja, najnižja in največja vrednost višine ter pokritost z vegetacijo.

LIDAR kot tehnika brezkontaktnega merjenja ima več prednosti. Najbolj pomembne so visoka natančnost, visoka gostota točk, velika pokritost območja in zmožnost hitrega in učinkovitega vzorčenja območja [19]. To ustvarja možnost preslikavanja diskretnih sprememb pri zelo visoki ločljivosti in enakomerno pokrivanje območij. Ostale prednosti so [27], da lahko izvajamo meritve podnevi ali ponoči in pri tem nismo izpostavljeni svetlobnim spremembam, kar poveča učinkovitost senzorja. Zelo pomembno je tudi, da – za razliko od drugih oblik zbiranja podatkov – tovrstni senzorji niso občutljivi na geometrijska popačenja ali izkrivljenja, kot so razgibane, kotno bogate pokrajine. Omenimo lahko tudi, da je laserski žarek bolj usmerjen od zvoka in da so laserski pregledovalniki primerni za natančne meritve, medtem ko so ultrazvočni merilniki zaradi mnogih vplivnih dejavnikov, ki so posledica lastnosti zvoka, podvrženi večjim negotovostim zaznavanja. Sisteme LIDAR je mogoče integrirati z drugimi viri podatkov, kar omogoča avtomatsko analiziranje kompleksnih podatkov.

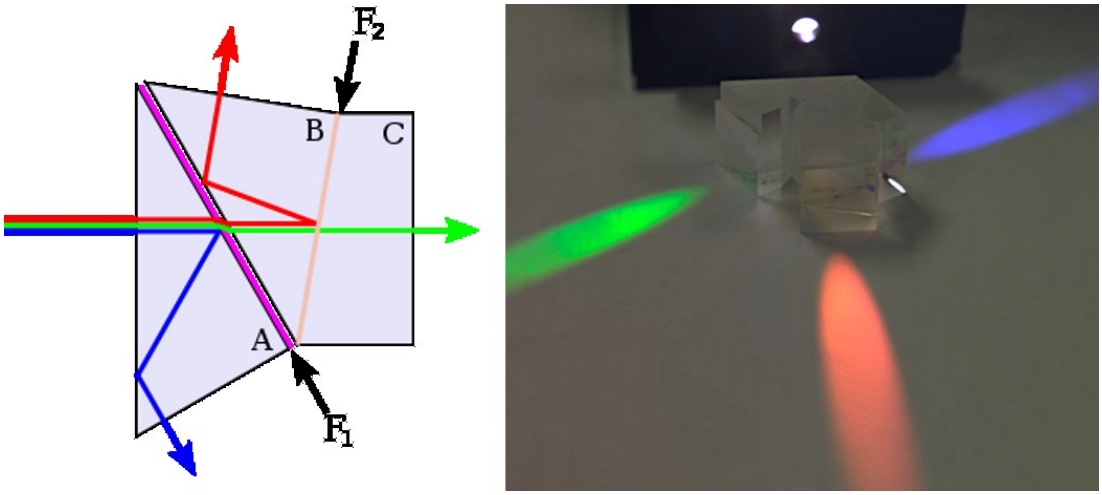
Poleg prednosti pa ima tudi slabosti. Na laserske žarke lahko vplivajo močen dež ali nizki oblaki zaradi učinkov loma, vendar se zbrani podatki še vedno lahko uporabijo za analizo. Ostale slabosti tehnologije LIDAR so povezane z višino na kateri lahko zajemamo meritve. Na višinah nad 2000 m ne more delovati dobro, ker impulzi niso učinkoviti na teh višinah [27]. Prav tako lahko zbiramo velike količine podatkov, ki zahtevajo visoko raven analize in interpretacije, za kar lahko porabimo veliko časa.

2.1.6 Kamera

Gre za optični inštrument, ki zajema slike ali snema videoposnetke in te informacije shranjuje na fizični medij digitalnega sistema ali pa na fotografski film [28]. Gre za preslikavo 3D-okolja v 2D-slikovni senzor. Če je slika digitalna, potem je sestavljena iz množice slikovnih elementov (angl. *pixels*).

Imamo več izvedb kamer glede na funkcionalnost, lastnosti in namen uporabe. V mobilnih sistemih uporabljamo navadno kamero, stereo kamero in ToF-kamero (angl. *Time-of-Flight camera*), ki temelji na merjenju časa preleta. Navadna kamera brez osvetlitve spada med pasivne senzorje. V poglavju 3.1 sta podrobneje obravnavani navadna kamera in stereo kamera.

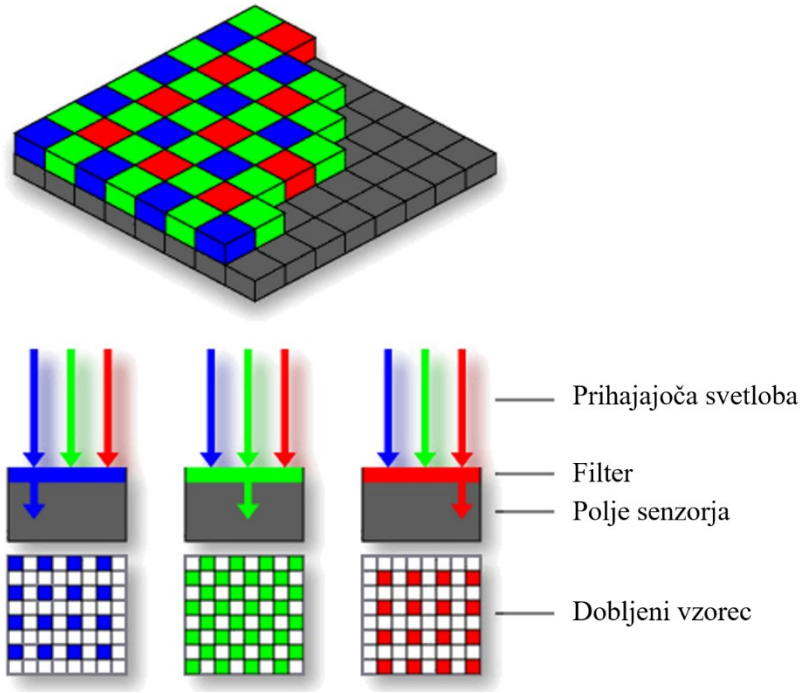
Mobilni sistem lahko s kamero razpoznava okolico in zaznava umetne ali naravne značilke. Kamera sestoji iz leče, ki usmeri svetlobo iz okolice na polje svetlobno občutljivih senzorjev (slikovni elementi slike) [2]. Vsak slikovni element izmeri jakost svetlobe, ki pade nanj. Končna meritev predstavlja jakost svetlobe posameznih slikovnih elementov. V primeru, da imamo barvno kamero, potrebujemo tri polja slikovnih elementov (za vsako barvno komponento barvnega prostora RGB svojo – izvedba s prizmo) ali pa polje z integriranimi filtri pred vsakim slikovnim elementom (Bayerjev filter), kot prikazujeta sliki 16 in 17.



Slika 16. Slikovni senzor s tremi CCD-senzorji (angl. *Charged Coupling Device*) (levo) in izvedba s prizmo (desno)

Leva fotografija: Cburnett [CC BY-SA 3.0] (https://en.wikipedia.org/wiki/Three-CCD_camera)
Desna fotografija: Dick Lyon (https://en.wikipedia.org/wiki/Three-CCD_camera)

Na sliki 17 zgoraj je prikazan primer Bayerjevega filtra, kjer se barvni filtri v običajnih sistemih nanašajo na eno plast fotodetektorjev v mozaičnem vzorcu. Filtri dopuščajo, da le ena valovna dolžina svetlobe (rdeča, zelena ali modra) preide skozi katerokoli slikovni element, kar omogoči zapis samo ene barve. Tipični mozaični senzorji zajamejo 50 % zelene in samo 25 % rdeče in modre svetlobe.



Slika 17. Nanos barvnega filtra na plast fotodetektorjev (zgoraj), prehod svetlobe z določeno valovno dolžino skozi določen slikovni element (sredina), prikaz zastopanosti rdeče, zelene in modre barve v filtru (spodaj)

Fotografija: Interiot [CC BY-SA 3.0] (https://commons.wikimedia.org/wiki/File:BayerPatternFiltration.png#globalusage)

V primeru digitalne kamere se slikovna informacija (napetostni nivo) vsakega slikovnega elementa še digitalizira.

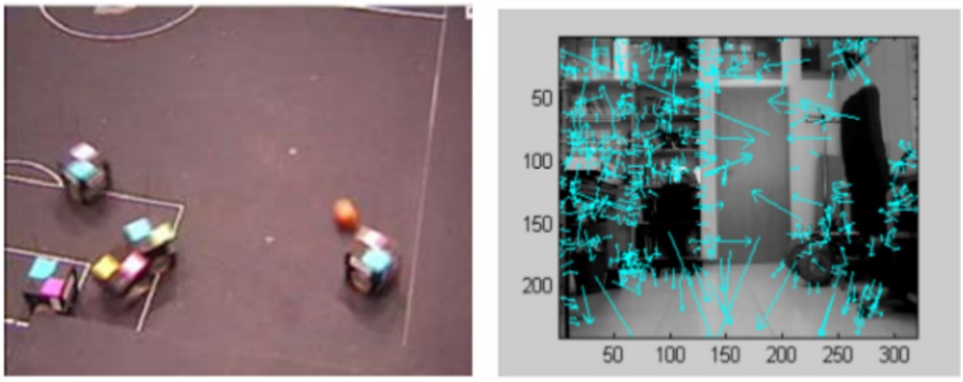
Meritev kamere je slika, ki je podana z zaporedjem števil, kjer je vsak slikovni element podan s trojico vrednosti (npr. barvna kamera z RGB ali drugim barvnim prostorom) ali z eno vrednostjo (npr. »črno-bela« kamera s sivinskimi odtenki) [2].

Kamere uporabljamo za:

- razpoznavanje objektov, oseb, avtomobilov,
- merjenje dimenzij objektov in relacij med njimi,
- kontrolo kakovosti proizvodov,
- zaznavanje okolja,
- lokalizacijo.

Slika oz. meritev kamere je informacijsko bogata, kar omogoča večjo prilagodljivost mobilnega sistema različnim namenom uporabe. Slikovne informacije so dobra podlaga za številne enostavne in kompleksne načine obdelave. Pod obdelavo razumemo iskanje elementov določene barve ali intenzitete, robov na sliki ali iskanje vnaprej znanih vzorcev z uporabo različnih generatorjev značilk, kot so SIFT (angl. *Scale Invariant Feature*), SURF (angl. *Speeded-Up Robust Features*), FAST (angl. *Features from Accelerated Segment Test*) in podobno [2]. Na sliki 18 so prikazani primeri obdelave slikovne informacije.

Slabost uporabe kamere je, da izgubimo informacijo o globini. Obdelava slikovne informacije pa je lahko računsko precej zahtevna operacija.

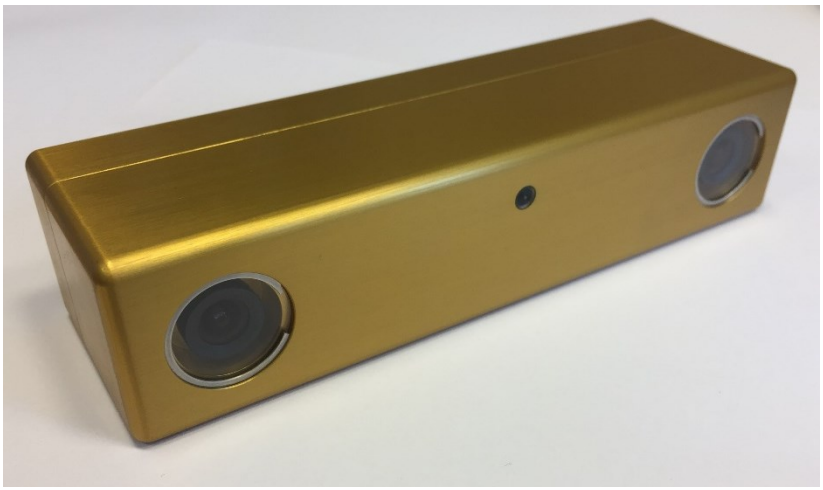


Slika 18. Detekcija robotov na osnovi barvne informacije (levo) in SIFT (angl. *Scale Invariant Feature Transforms*) značilke (desno)

2.2 Stereo kamera

»Kamera preslika tridimenzionalni prizor iz okolice v dvodimenzionalno sliko na polju slikovnih senzorjev. Pri tej preslikavi se izgubi informacija o globini. Da iz slike kamere rekonstruiramo globinsko informacijo, sta potrebni dve sliki istega prizora, posneti iz različnih

leg.« (Klančar, 2014, str. 130) Do rekonstrukcije 3D-točke lahko pridemo z modelom stereo kamere, ki je opisan v poglavju 3.1. Na sliki 19 je prikazana stereo kamera Bumblebee.



Slika 19. Stereo kamera Bumblebee proizvajalca Point Grey

Uporabo stereo kamere srečamo v industriji zabave, pri prenosu informacij in avtomatiziranih sistemih (npr. v proizvodnji oz. kontroli kakovosti). Stereo vid je pomemben na področju kot je robotika, zaradi zmožnosti pridobivanja relativnih položajev in prepoznavanja tridimenzionalnih objektov. Stereo kamere se uporabljajo pri detekciji vozil v stvarnem času [29], kot asistenca pri vožnji z avtomobilom [30], za zaznavanje in sledenje vozil [31].

Stereo kamere uporabljamo v zunanjem in notranjem okolju. Poznamo aktivne in pasivne sisteme, ki so predstavljeni v podpoglavjih 2.2.1 in 2.2.2.

2.2.1 Pasivni sistemi

Pasivni stereo vid, kot sistem dveh ali več senzorjev, ne potrebuje oddajati lastne energije ali osvetlitve v okolje za izvedbo meritev. Za 3D-rekonstrukcijo oblike je potrebno rešiti zahteven problem iskanja ustreznih parov točk med različnimi pogledi. Ta problem lahko omilimo z uporabo kontroliranega izvora strukturirane svetlobe [32] – to je potem že konvencionalni aktivni stereo vid.

Primer uporabe pasivnega sistema stereo kamere je v avtonomnih pristajalnih sistemih za namen identifikacije varne pristajalne cone [24], kjer je stereo kamera nameščena na krila zrakoplova.

2.2.2 Aktivni sistemi

Aktivni stereo vid aktivno uporablja svetlobo, kot je laser ali strukturirana svetloba, da poenostavi problem stereo ujemanja. Strukturirana svetloba pomeni projiciranje znanega

vzorca na sceno. Izvor svetlobe je v tem primeru projektor, ki mora biti kalibriran. Poznamo več načinov uporabe svetlobe v aktivnih sistemih [33]:

- Konvencionalni vid s strukturirano svetlobo (angl. *conventional structured-light vision*) [34], ki uporablja strukturirano svetlobo ali laser in išče korespondence projektor-kamera.
- Konvencionalni aktivni stereo vid (angl. *conventional active stereo vision*) [35], ki prav tako uporablja strukturirano svetlobo ali laser, medtem ko stereo ujemanje izvajamo za korespondence kamera-kamera na enak način kot pri pasivnem stereo vidu.
- Stereo s strukturirano svetlobo (angl. *structured-light stereo*) [35], ki vpelje hibridno tehniko, ki uporablja ali združuje korespondence kamera-kamera in projektor-kamera.

2.3 Sistemi za izogibanje oviram

V tem poglavju bomo predstavili sisteme za izogibanje oviram. Opisali bomo zgradbo senzorskega sistema in metode izogibanja oviram ter ostale funkcionalnosti tovrstnih avtonomnih mobilnih sistemov.

2.3.1 Primer robotskega sesalnika

Roomba je serija avtonomnih robotskih sesalnikov (slika 20), ki jih proizvaja podjetje iRobot.

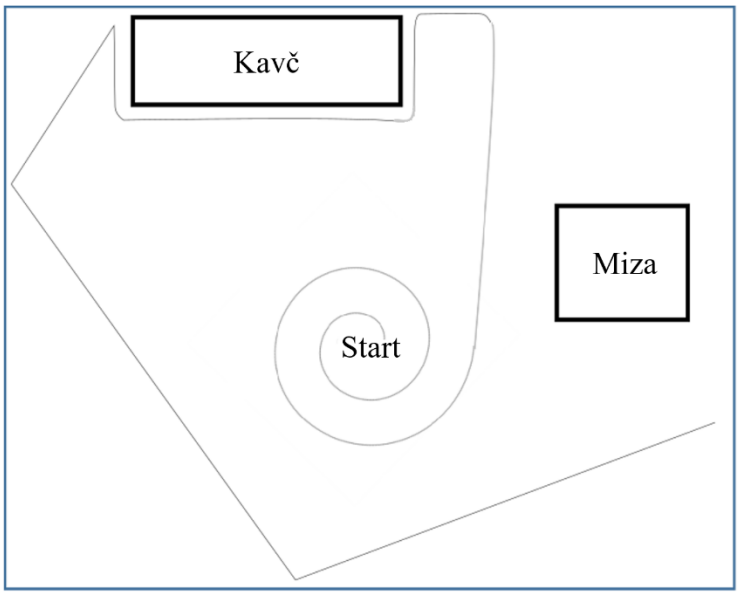
Slika 20. Roomba 870 (levo), Roomba 980 (desno)

Leva Fotografija: Kārlis Dambrāns [CC BY 2.0] (https://www.flickr.com/photos/janitors/16048185255)
Desna Fotografija: weird.com [CC BY-SA 4.0] (https://commons.wikimedia.org/wiki/File:IRobot_Roomba_980.jpg)

Avtonomni robotski sesalnik deluje v več načinih: naključno gibanje in način sledenja steni [36]. V naključnem kontaktnem načinu se robot giblje po ravni liniji, dokler ne pride v kontakt z oviro. Robot nato zavije stran od ovire in nadaljuje v novi naključni smeri. V načinu sledenja steni mora robot zaznati steno, ki ji nekaj časa sledi in se nato vrne v naključni način delovanja oz. gibanja po prostoru. S kombinacijo teh dveh načinov vožnje po prostoru so

izumitelji patenta [37] pokazali, da robotski sesalnik pokrije ustrezen del tal, vključno z robovi, v optimalnem času. To je pomembno predvsem zaradi prihranka energije.

Mobilni robot Roomba ima nadzorni sistem, s katerim učinkovito pokriva dano območje z delovanjem v več načinih, vključno s sledenjem oviri in naključnim iskanjem. V drugih izvedbah, Roomba zagotavlja pokritost točk prostora s spiralnim gibanjem in drugimi načini [37]. Na sliki 21 je prikazan potek čiščenja robotskega sesalnika, ki ga je opisal avtor v prispevku [38]. Roomba začne čiščenje v spirali, ki se premika navzven, nato pa se usmeri po obodu sobe. Ko robotski sesalnik zadene oviro, predpostavi, da je dosegel obod sobe. Nato čisti po obodu, dokler ne zadene druge ovire, ki ji sledi. Nato poišče čisto pot in nadaljuje z različnimi kombinacijami gibanja po prostoru.



Slika 21. Algoritem čiščenja

Cilj sistema zaznavanja ovir je navigacija v prostoru glede na površine v okolici z uporabo primernega senzorskega podsistema. V času razvoja so se senzorski podsistemi, kot so ultrazvočni senzorji oz. senzorji, ki temeljijo na principu sonarja, izkazali za predrage ali preveč kompleksne pri detekciji ovir na tleh, detekciji sten, da bi lahko prešli v način sledenja steni (ali da bi se izognili zaletavanju v steno) [36]. Taktilni senzorji pa so se izkazali za neučinkovite. Robotski sistem je za zaznavanje ovir potreboval sistem, ki je preprost v zasnovi, poceni, natančen, enostaven za izvedbo in enostaven za umerjanje.

Na senzorski podsistem ne smejo vplivati različne odbojnosti površin, zato so se izumitelji odločili za uporabo infrardečega senzorja pri zaznavanju sten in ovir. Infrardeči senzor vključuje optični oddajnik, ki oddaja usmerjen infrardeči žarek z določenim območjem sevanja v okolico in fotonski sprejemnik, ki ima določeno vidno polje. Območje sekanja obeh polj določa zaznano okolico, s katero lahko določimo območje ovire, stopnice ali stene in s tem konstruiranje primernih ukazov robotu za izogibanje oviram ali drugemu načinu obnašanja v

trenutnem okolju. Za zaznavanje stopnic ali strmih padcev ima robotski sesalnik spodaj nameščene dodatne infrardeče senzorje, ki so usmerjeni naravnost navzdol. Poleg infrardečega senzorja se na sprednjem delu robota nahaja odbijač s senzorjem dotika. Z infrardečimi žarki zaznamo ovire in stene, kar robotski sesalnik izkorišča za upočasnjevanje pred ovirami. Ko robot naleti na oviro, jo zazna s prednjim odbijačem in se ustavi.

Pomembna je tudi oblika samega robota, ki je okrogel in nizek, kar mu omogoča, da potuje pod pohištvom. Zaradi okrogle oblike robota so določeni algoritmi bolj enostavni za implementacijo in se lahko vrti na mestu, kar olajša gibanje oz. spreminjanje smeri, ko naleti na oviro.

Da bi lahko zaznali bolj umazane dele tal, robotski sesalnik uporablja t. i. piezoelektrični senzor [39]. To je v osnovi kristal, ki proizvaja električne impulze, ko pride v fizični kontakt z drugimi predmeti. Ko delci umazanije zadenejo senzor, ta proizvede majhne električne impulze, katerih pretirano število domnevno naznani, da smo naleteli na pretežno umazan vzorec tal. Tako lahko robot bolj temeljito očisti umazane predele.

Prve verzije robotskega sesalnika Roomba so čistile prostore skoraj povsem naključno, brez sprotnega grajenja zemljevidov prostorov. Zato so za čiščenje potrebovali veliko časa. Novejše verzije sesalnika so napredovale k inteligentnejšem pristopu kot je sočasna lokalizacija in gradnja zemljevida na osnovi slike VSLAM (angl. *Visual Simultaneous Localization and Mapping*). Robotski sesalnik ima vgrajeno kamero, s katero naredi posnetek prostora in postopoma sestavlja zemljevid prostora, tako da ve, kam gre in kje je že bil. Takšen robotski sesalnik lahko hitreje čisti in se giblje bolj sistematično. Gradnja zemljevida pomeni vključevanje ovir, sten in ostalih predmetov, ki ji zaznamo s senzorskim sistemom s prepoznavanjem značilnosti posnete okolice.

Metoda upravljanja mobilnega robota vključuje generiranje segmentiranega zemljevida, ki definira posamezna področja površine na podlagi podatkov o zasedenosti, ki jih zbira mobilni robot. Omogočeno je prepoznavanje podobmočij in računanje vzorca pokritosti prostora glede na identifikacijo podobmočij na tista, ki jih je potrebno očistiti, in tista, ki so že bila očiščena. Vzorec pokritosti služi robotu za organizacijo gibanja po obdelanem in neobdelanem prostoru, tj. čistem in morda umazanem območju.

2.3.2 Primer raziskovalnega mobilnega sistema

Raziskovalni mobilni sistem Pioneer 3-AT je vsestranska štirikolesna robotska platforma (slika 22), kar pomeni, da je mobilni robot primeren za mnoge aplikacije v notranjem in zunanjem prostoru ter terenskih projektih. Robot je močen, enostaven za uporabo, zanesljiv in fleksibilen. Platforma ponuja možnost vgrajenega računalnika, kar ponuja odprte možnosti za procesiranje računalniškega vida, podatkov z laserja, podatkov z diferencialnega GPS-sistema in ostale avtonomne funkcije.

Slika 22. Robot Pioneer 3-AT

Robot je opremljen z različnimi senzorji: ultrazvočnimi senzorji razdalje, laserskim pregledovalnik razdalj (SICK LMS 200), stereo kamero, kompasom in inkrementalnimi merilniki zasuka na kolesih.

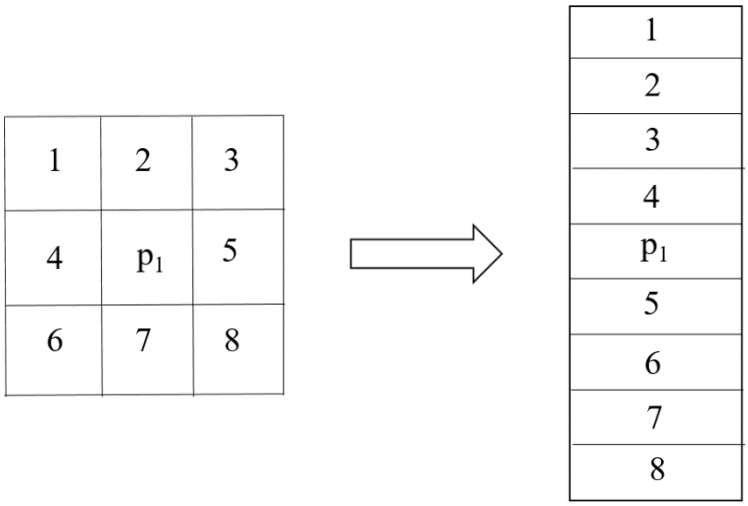
V nadaljevanju je opisan postopek za izogibanje oviram na podlagi stereo vida, ki so ga predstavili avtorji v članku [40]. Zaznavanje ovir je narejeno s segmentacijo slike in algoritmom stereo vida, ki lahko loči ovire od ozadja in poišče stereo ujemanja njihovih značilnosti oz. konture z rezultati kalibracije stereo vida. Na ta način lahko pridobimo prostorske točke za 3D-rekonstrukcijo.

Za iskanje kontur ovir je v tem primeru najprej potrebno sliko v RGB-formatu spremeniti v sivinsko sliko. Nato so uporabljene morfološke operacije za odstranjevanje šuma, izolacijo in združevanje posameznih elementov ter za iskanje ovir, ki predstavljajo intenzivne luknje ali prepade na slikah. Po uporabi morfološke operacije odpiranja je večkrat uporabljena operacija zapiranja, s čimer lahko jasno ločimo območje ovire od ozadja. Končno lahko obliko konture pridobimo z uporabo algoritma Canny za detekcijo robov. Lastnost algoritma Canny je, da poskuša posamezne kandidate slikovnih elementov slike, ki predstavljajo robne točke, sestaviti v konture. To pomeni, da imamo dva praga, zgornji in spodnji, ki ju je potrebno določiti skozi več eksperimentov. Eksperimenti so pokazali tudi, da je možno razpoznati konture ovir do oddaljenosti treh metrov.

Stereo ujemanje značilnosti je narejeno po metodi okenc velikosti NxN. Cilj metode je najti točke $\boldsymbol{p}_1 = (x_1, y_1, 1)^T$ in $\boldsymbol{p}_2 = (x_2, y_2, 1)^T$, ki se ujemajo med levo in desno stereo popravljeno sliko pridobljene s kalibrirano kamero. Popravljanje slike je proces popravljanja posameznih slik, kar pomeni da je končna oblika slik taka, kot da bi jih posneli z dvema kamerama z vrstično poravnanima ravninama slik. To pomeni, da sta optični osi dveh kamer vzporedni. Popravljanje slik je v tem primeru narejeno z uporabo rotacijskih in translacijskih matrik z algoritmom, ki ga je predstavil Bouguet [41]. Ta algoritem poskrbi, da sta slikovni

ravnini obeh kamer poravnani. Iskanje korespondenc med levo in desno sliko je posledično lažje, saj so vrstice slike po postopku popravljanja poravnane in tako zmanjšamo napako pri stereo ujemanju, predvsem pa pohitrimo iskanje parov.

Točki $\boldsymbol{p}_1$ in $\boldsymbol{p}_2$ predstavljata pri procesu ujemanja središča okenca (slika 23). Vse točke v okencu, kjer je točka $\boldsymbol{p}_1$ središče, tvorijo vektor $\boldsymbol{v}_1$. Enako velja za vektor $\boldsymbol{v}_2$.



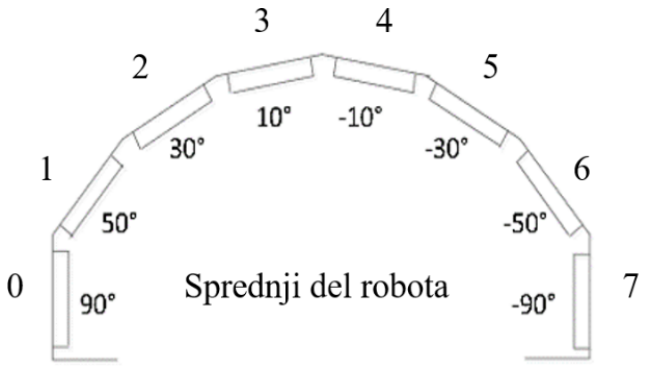
Slika 23. Okence sivinskih vrednosti slike okoli središča (levo), pripadajoči vektor (desno)

Manjši kot je kot med vektorjema, večja je stopnja ujemanja med točkama $\boldsymbol{p}_1$ in $\boldsymbol{p}_2$. Kot med vektorjema računamo po enačbi

$$\cos\theta_v = \frac{\boldsymbol{v}_1{}^T \boldsymbol{v}_2}{\|\boldsymbol{v}_1\|\|\boldsymbol{v}_2\|}. \tag{3}$$

Za vsako značilnost v levi sliki se v desni sliki išče ustrezna vrstica za najboljše ujemanje z algoritmom ujemanja na principu okenca. Po rektifikaciji oz. popravljanju slike je vsaka vrstica epipolarna črta, zato mora biti ujemajoča se točka $\boldsymbol{p}_1$ z največjim kotom na desni sliki vzdolž iste y-koordinate kot točka $\boldsymbol{p}_2$ na levi sliki. Večja kot je dispariteta, tem bližja je razdalja. Rezultat opisanega postopka je največja dispariteta slike.

Metoda za izogibanje oviram uporablja stereo vid in ultrazvočne senzorje, ki delujejo vzajemno za pridobitev informacije o ovirah v bližini robota. Ultrazvočni senzorji so uporabljeni za kompenzacijo mrtvega kota oz. točk kjer stereo kamera ne uspe pridobiti informacije zaradi omejenega vidnega kota. Razporeditev ultrazvočnih senzorjev je prikazana na sliki 24 in prikazuje območje, ki ga pokrivajo.

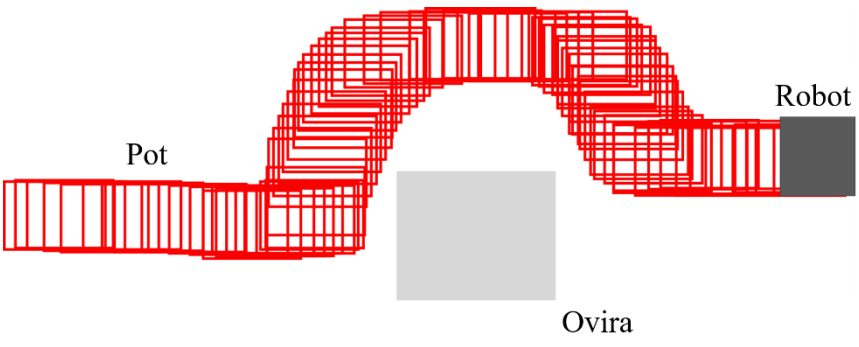


Slika 24. Obroč ultrazvočnih senzorjev robota Pioneer 3-AT

Algoritem vodenja temelji na principu mehke logike in je uporabljen za izogibanje trkom ter da lahko ovire obidemo. Sistem vodenja vsebuje tri vhode in dva izhoda. Območje zaznavanja pri robotu je razdeljeno na tri dele: sprednji, sprednji-levi in sprednji-desni del. Ultrazvočna senzorja nič in ena sta izbrana, da priskrbita razdaljo v sprednjem-levem delu, medtem ko sta senzorja šest in sedem zadolžena za sprednji-desni del. Razdaljo do ovir v sprednjem delu pa pridobivamo s stereo kamero. Najmanjša vrednost vsakega dela je izbrana kot vhodni parameter v sistem vodenja.

Vhode spremenljivke mehkega krmilnika so x_1, x_2, x_3 in predstavljajo razdalje do ovir v treh območjih merjenja. Izhoda krmilnika sta y_1 in y_2, ki predstavljata levo in desno hitrost kolesa. Ti hitrosti pa določata dve obliki gibanja, in sicer linearno ali krožno gibanje robota.

Mehčanje vhodnih in izhodnih spremenljivk je določeno s trikotno pripadnostno funkcijo. Za ostrenje izhodnih podatkov je uporabljena metoda težišča. Slika 25 prikazuje rezultate simulacije vodenja oz. izogibanja oviram. Vidimo da se je robot najprej peljal naravnost, ko je zaznal oviro jo je obšel in nadaljeval pot.



Slika 25. Simulacija izogibanja oviram

2.3.3 Primer avtonomnega avtomobila

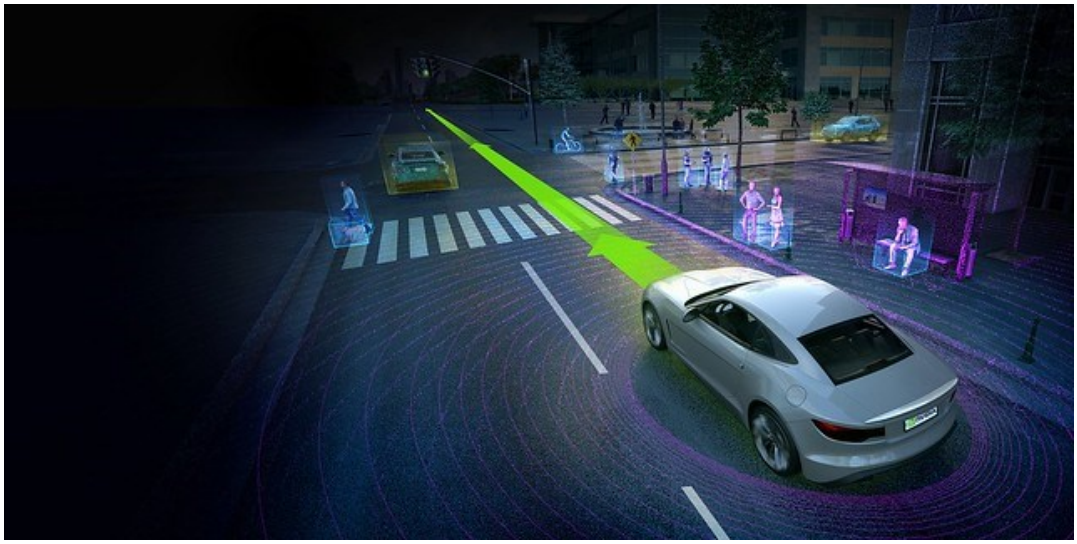
Tesla je ime, ki si ga delijo električni avtomobili ameriškega avtomobilskega in energetskega podjetja. Od marca 2019 prodajajo prenovljen seznam semiavtonomnih modelov, in sicer Model S, Model 3, Model X, Model Y in športni avtomobil Roadster. Ker je tehnologija enaka in uporabljena v vseh Teslinih vozilih z izjemo vozila Roadster, jo bomo opisali splošno ne glede na tip modela ali vozila.

Vsi novi Teslini avtomobili imajo potrebno strojno opremo za popolno avtonomno vožnjo v prihodnosti v skoraj vseh okoliščinah. Sistem je zasnovan tako, da lahko opravlja potovanja na kratke in dolge razdalje, pri čemer osebi na voznikovem sedežu ni potrebno sodelovati v procesu vožnje [42].

Vendar pa trenutne funkcije t. i. Teslinega avtopilota zahtevajo aktiven nadzor voznika in vozila ne naredijo avtonomnega. Avtopilot omogoča avtomobilu, da avtomatično zavija, pospešuje in zavira znotraj voznega pasu. Varnostne in asistenčne funkcije [43], ki jih ima vsako vozilo so:

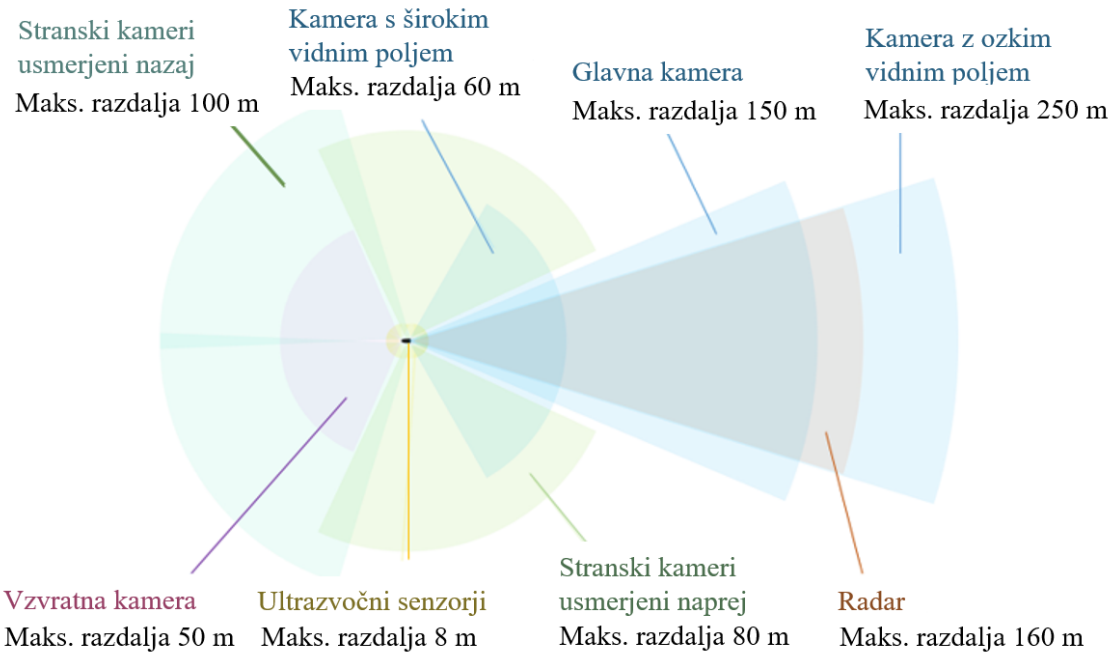
- izogibanje trkom,
- avtomatično zaviranje v sili (zasnovan tako, da zazna predmete, ki ji avtomobil lahko zadene in ustrezno izvede proces zaviranja),
- opozorilo o stranskem trku (voznika opozori na morebitna trčenja z ovirami ob avtomobilu),
- opozorilo o prednjem trku (opozori na bližajoče trke s počasnejšimi ali stacionarnimi avtomobili),
- avtomatično prilagodljive dolge luči,
- adaptivni tempomat,
- avtomatično upravljanje volana,
- menjava voznega pasu (ko voznik nakaže smer menjave s smerniki),
- navigacijski sistem (avtomatična navigacija do želene destinacije) in
- t. i. *Summon* način na poziv (premikanje avtomobila na daljavo preko ključa ali pametnega telefona, npr. iz parkirnega prostora).

Tako imenovani Tesla Vision [42], zgrajen na globokem nevronskem omrežju, segmentira okolje avtomobila z več stopnjami zanesljivosti kot tista, ki jih dosežemo s klasičnimi tehnikami obdelave vida. Primer obdelave podatkov je prikazan na sliki 26, na kateri vidimo razpoznavanje ovir (pešcev in avtomobilov) na cesti.



Slika 26. Primer zaznavanja objektov v prometu

Tesla Vision temelji na različnih izvedbah osmih kamer, ki so vgrajene zadaj, spredaj in na straneh ter zagotavljajo 360° vidljivost do razdalje 250 m. Senzorski sistem je poleg kamer sestavljen še iz ultrazvočnih senzorjev in radarja kot je prikazano na sliki 27.



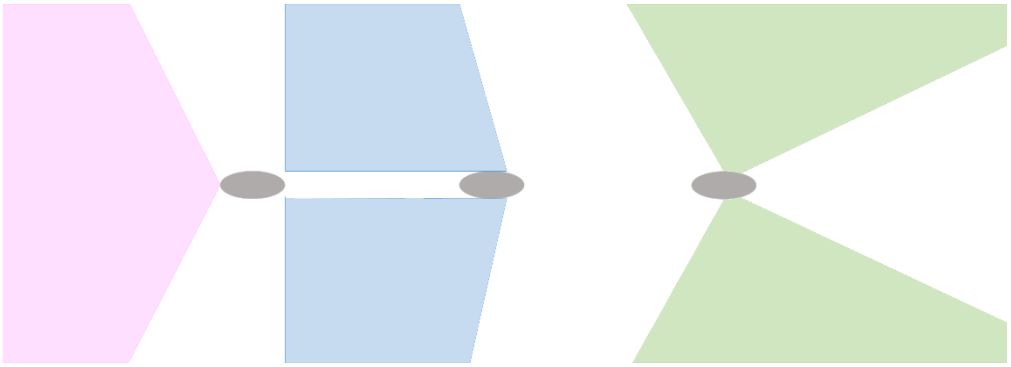
Slika 27. Senzorski sistem Tesle

Tri kamere, s specifičnimi nalogami, so nameščene za vetrobranskim steklom in zagotavljajo široko vidljivost pred avtomobilom in osredotočeno zaznavanje oddaljenih predmetov [42]. Te kamere se razlikujejo glede na kot vidnega polja.

Kamera s širokim vidnim poljem ima 120-stopinjsko lečo ribjega očesa in lahko zajame semaforje in ovire, ki segajo na pot, ter predmete v neposredni bližini, kar je posebej uporabno

pri mestnem manevriranju z majhnimi hitrostmi. Ta kamera ima najmanjši doseg od ostalih treh. Za razliko od širokokotne ima glavna kamera ožje vidno polje, vendar pa ima daljši doseg zaznavanja, kar je prikazano na sliki 27. T. i. ozkokotna kamera pa omogoča pogled na velikih razdaljah na oddaljene značilke, kar je uporabno pri manevrih pri velikih hitrostih.

Preostale kamere prikazane na slikah 27 in 28 so: vzvratna kamera, stranski kameri usmerjeni nazaj in stranski kameri usmerjeni naprej.

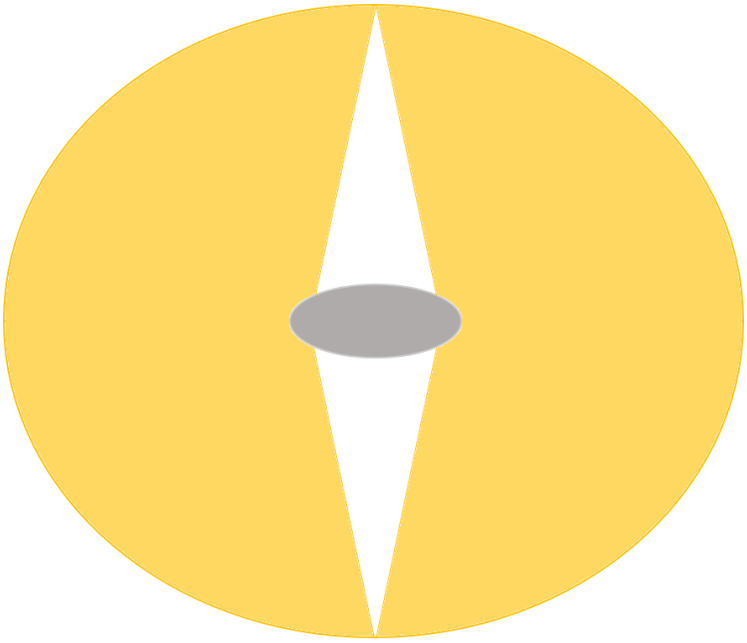


Slika 28. Vzvratna kamera (levo), stranski kameri usmerjeni nazaj (na sredini), stranski kameri usmerjeni naprej (desno)

Vzvratna kamera ni koristna le za varno vzvratno vožnjo, temveč daje znaten prispevek k Teslinemu avtopilotu. 90-stopinjske stranske kamere, usmerjene naprej, iščejo avtomobile v sprednjem delu vozila oz. vidnem polju, ki nepričakovano vstopajo na vozni pas vozila na avtocesti in zagotavljajo dodatno varnost pri vstopu na križišča z omejeno vidljivostjo. Stranske kamere usmerjene nazaj spremljajo mrtve kote v zadnjem delu avtomobila na obeh straneh, kar je pomembno pri varni menjavi voznih pasov in vključitvi v promet.

Radar na sprednji strani vozila (slika 27) zagotavlja zaznavanje predmetov na razdalji do 160 m. Z radarsko tehnologijo lahko zaznavamo predmete skozi močan dež, meglo in prah. Lastnosti radijskega valovanja z določeno valovno dolžino omogočajo tudi prehajanje skozi oz. pod avtomobil.

V sistem je vključenih tudi 12 ultrazvočnih senzorjev, ki zaznavajo prihajajoče avtomobile do razdalje osmih metrov, preprečujejo možne trke in služijo kot pomoč pri parkiranju. Na sliki 29 je prikazano območje zaznavanja ultrazvočnih senzorjev.



Slika 29. Območje zaznavanja ultrazvočnih senzorjev

Osem kamer in 12 ultrazvočnih senzorjev zaznava črte voznega pasu in okoliške predmete z zagotovljeno 360° vidljivost v vsakem trenutku (slika 27). Vsi Teslini modeli se ohranjajo v voznem pasu, pri čemer prilagajajo hitrost prometnim razmeram brez voznikovega poseganja v proces vožnje.

Drugi primer avtonomnega vozila je samovozeče vozilo Waymo (prikazano na sliki 30), ki ima začetke v letu 2009 kot Google-ov projekt.

Slika 30. Vozilo Waymo

Fotografija: Dllu [CC BY-SA 4.0] (https://en.wikipedia.org/wiki/Waymo)

Senzorski sistem vozil Waymo je sestavljen iz kamer, LIDAR-ja in radarja [44]. Visoko ločljiva kamera zaznava vizualno informacijo (npr. ali je luč na semaforju zelena ali rdeča). Z uporabo radarja lahko zaznamo, kako daleč so objekti in izračunamo njihovo hitrost. Senzor LIDAR

pošilja milijone laserskih žarkov v okolje vsako sekundo, da bi lahko zgradili podrobno 360° sliko sveta okoli sebe.

Senzorji in programska oprema nenehno pregledujejo predmete okoli vozila: pešce, kolesarje, vozila, dela na cesti in ovire [45]. Waymo neprestano bere nadzor prometa, prometne znake, barvo luči semaforja, železniške zapornice, vse do začasnih stop znakov. Vozila Waymo lahko vidijo daleč v velikosti treh nogometnih igrišč v vseh smereh. Programska oprema predvideva premike vsega okoli nas na podlagi njihove hitrosti in poti. Avtonomno vozilo Waymo razume, da se vozila gibljejo drugače kot kolesarji ali pešci. Na podlagi teh podatkov programska oprema napoveduje možne poti ostalih udeležencev v prometu.

Waymo združuje podatke, da bi lahko razumel svet okoli sebe. V vsakem trenutku ve, kje se nahaja in predvideva, kaj bodo zaznani predmeti storili v prihodnosti. Poleg ločevanja udeležencev v prometu na avtomobile in pešce, Waymo načrtuje pot vnaprej.

Torej na podlagi vseh teh informacij avtonomno vozilo Waymo določa natančno smer, hitrost, vozni pas in krmilja, potrebna za varno in avtonomno vožnjo.

2.3.4 Primer avtonomnega letalnika

Skydio R1 je avtonomni letalnik, ki združuje umetno inteligenco, računalniški vid in napredno robotiko [46]. Zgrajen je za popolno avtonomijo, ki temelji na vizualnem zaznavanju. Letalnik R1 ima sposobnost, da leti, medtem ko avtonomno sledi objektu in se izogiba trkom ter snema 4K-posnetke. Uporabnik ima možnost nadzora letalnika preko mobilne aplikacije. Letalnik R1 zajema stabiliziran 4K-video z uporabo 3-osne stabilizacije in naprednega slikanja. Na sliki 31 je prikazan primerek letalnika proizvajalca DJI.

Slika 31. Primerek letalnika proizvajalca DJI

Fotografija: Simon Waldherr [CC BY-NC-SA 2.0] (https://www.flickr.com/photos/simonwaldherr/39810249765)

Senzorski sistem R1 je sestavljen iz trinajstih kamer, od tega je dvanajst namenjenih navigaciji in ena za snemanje po želji uporabnika. V sistem so vključeni tudi GPS, barometer in štiri inercialne merilne enote.

Letalnik R1 uporablja tehnologijo *Skydio Autonomy Engine*, ki je bila tema raziskav in razvoja pri podjetju Skydio. *Skydio Autonomy Engine* razume, kaj se dogaja okoli letalnika in predvideva, kaj se bo zgodilo ter sprejema odločitve večkrat na sekundo. Uporablja 13 kamer za izdelavo 3D-zemljevida okolice, ki vključuje drevesa, ljudi, zgradbe in še več. V trenutku, ko je v zraku, se letalnik začne izogibati oviram, tudi skozi zahtevna okolja, kot je npr. gozdna pot. Letalnik lahko leti sam in uporabnik ne potrebuje posebnih znanj in sposobnosti za upravljanje z letalnikom. Letalnik uporablja svoj napredni vidni sistem, da vas spremlja v 3D-prostoru, ko se gibljete.

Obnašanje letalnika Skydio R1 nadziramo preko aplikacije na mobilnem telefonu. Aplikacija Skydio uporablja povezavo WiFi pametnega telefona za komunikacijo z letalnikom in omogoča prenos prizora v živo, ki ga trenutno zajema. V aplikaciji lahko preko enostavnega in intuitivnega vmesnika izbiramo različne načine delovanja, ki ustrezajo uporabnikovim aktivnostim. Letalnik lahko uporabi GPS-signal pametnega telefona, da uporabnika ponovno zazna in mu začne slediti ali se po potrebi vrne k njemu. V tabeli 2 so opisane spretnosti oz. načini delovanja letalnika Skydio R1 [47], ki jih uporabnik lahko izbere v aplikaciji.

Tabela 2. Načini delovanja letalnika Skydio R1

Spretnost/način	**Opis**
Bumerang	Poleti nazaj in razkrij pokrajino pred osebo.
Kabelska kamera	Premikaj se gladko med dvema točkama vzdolž navideznega kabla. Po izbiri opazuj ali sledi osebi med gibanjem po kablu.
Sledenje avtomobilu	Spremljaj in sledi avtomobilu.
Avtomobilski trinožnik	Ostani na miru in spremljaj avtomobil.
Kompas	Ohranjaj fiksno smer kompasa med gibanjem.
Zabavno	Leti gor in vstran in pri tem opazuj izbrano osebo.
Sledenje osebi	Sledi osebi za hitre akcije.
Igralna palica	Standardna igralna palica (ročni let z izogibanjem oviram).

Let v vodstvu	Leti pred osebo glede na njeno gibanje.
Kroženje	Kroženje okoli osebe.
Četrtinsko sledenje	Ostani med stransko in zadnjo stranjo osebe.
Četrtinsko sledenje v vodstvu	Ostani med stransko in prednjo stranjo osebe.
Raketa	Leti naravnost navzgor med gledanjem osebe navzdol.
Poleg	Glede na gibanje osebe ostani ob njeni strani.
Gladko	Natančno sledi osebi za kinematografski video (to omogoča, da se objekt premika bolj v kadru, ne da bi se letalnik pogosto prilagajal)
Stadion	Za šport na terenu ali podobne dejavnosti ostani visoko nad akcijo.
Trinožnik	Bodi visoko nad osebo in ostani stacionaren na enem mestu, medtem ko se kamera premika, da se usmeri proti osebi.
Vrtinec	Leti navzgor in stran od osebe.

Pomanjkljivosti letalnika so, da ni vodoodporen in zahteva dobro vidljivost in ne sme leteti v neugodnih vremenskih razmerah, kot so dež, megla ali dež [48]. Veter omejuje delovanje letalnika in ne sme leteti, ko vetrovi oz. sunki vetra presegajo hitrosti približno 19 $\frac{km}{h}$.

Letalnik R1 ne more videti ali razpoznati določenih ovir. Pazljivi moramo biti, ko leti okrog tankih vej, daljnovodov, vrvi, žic, transparentnih površin (okna) in odbojnih površin (ogledala, mirujoča voda). Če letimo z letalnikom R1 v bližini teh ovir lahko povzročimo poškodbe na letalniku zaradi možnih trkov.

Letalnik R1 lahko naenkrat sledi samo eni osebi. Drugih ljudi, še posebej, če so v gibanju, morda ne vidi dovolj hitro, da bi se jim R1 izognil varno. Najvarnejši način, kako se

izogniti drugim ljudem, je letenje nad njimi. Če je naokoli veliko ljudi, se lahko izognemo morebitnim trkom z delovanjem v načinu *stadion*.

Letalnik R1 se pri pristanku ne izogiba oviram. Ko na telefonu izberemo možnost, da letalnik pristane, se ta spusti naravnost navzdol. Zato je potrebno, da poskrbimo, da je pristajalno območje ravno in brez ovir. Z letalnikom lahko letimo do hitrosti 40 $\frac{km}{h}$ v skoraj vseh situacijah, kar pomeni, da lahko sledimo pešcem, kolesarjem, smučarjem in ostalim.

3 Zaznavanje ovir s stereo kamero

To poglavje je osredotočeno na zaznavanje ovir s pomočjo stereo kamere, bolj natančno z globinsko kamero RealSense D435i proizvajalca Intel. V poglavju 3.1 je predstavljen model stereo kamere in način rekonstrukcije tridimenzionalnega položaja točke. Predstavljen je tudi perspektivični model kamere, opisane so distorzije in aberacije, ki so lahko posledica nepopolne izdelave optičnih elementov (npr. leče) ali tudi določenih karakteristik okolja. V poglavju 3.2 je predstavljeno merjenje globine s stereo kamero, kjer smo se osredotočili na globinsko kamero D435i, ki je sestavni del praktičnega dela magistrske naloge. Predstavljen je pomen kalibracije kamere, opisane so metode iskanja parov, način izračuna disparitete, izračun globine in funkcije postprocesiranja ter alternativni pristopi pri pridobivanju globinske informacije iz okolja. V poglavju 3.4 je predstavljen problem zaznavanja ovir v oblaku točk.

3.1 Perspektivični model kamere

Projekcija je transformacija prostora z $N > 0$ dimenzijami v prostor z $M < N$ dimenzijami [1]. Pri splošni projekcijski transformaciji se nekaj informacije izgubi. Če je možna večkratna projekcija objekta, potem je možno v nekaterih primerih rekonstruirati originalni objekt v *N*-dimenzionalnem prostoru. Najbolj pogosti sta perspektivična in paralelna projekcija, pri čemer je slednja linearna transformacija, ki ima žariščno točko v neskončnosti.

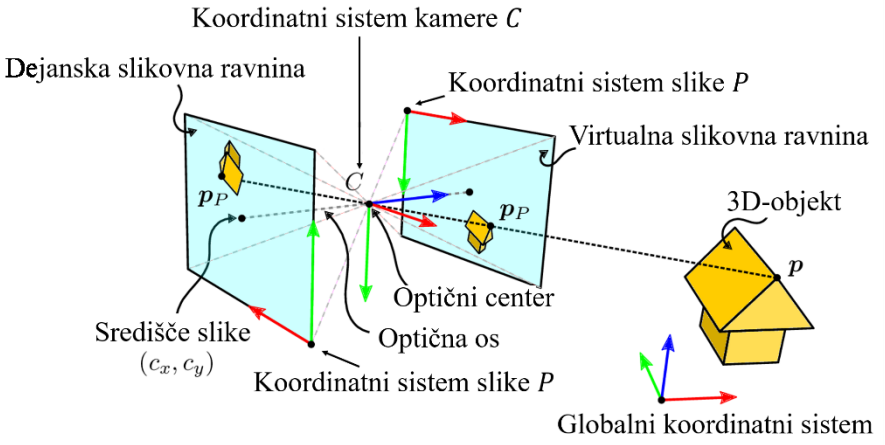
Glede na model kamere z luknjico (angl. *pinhole camera model*) se tridimenzionalna točka $\boldsymbol{p}_C^T = [x_C\ y_C\ z_C]$, podana v koordinatnem sistemu kamere C, projecira v dvodimenzionalno točko $\boldsymbol{p}_P^T = [x_P\ y_P]$ v koordinatnem sistemu slike *P* (slika 32) z enačbo

$$\underline{\boldsymbol{p}}_P = \frac{1}{z_C}\boldsymbol{S}\boldsymbol{p}_C, \tag{4}$$

kjer je $\underline{\boldsymbol{p}}_P$ predstavitev točke $\boldsymbol{p}_P$ v homogenih koordinatah, tj. $\underline{\boldsymbol{p}}_P{}^T = [x_P\ y_P\ 1]$. Matrika $\boldsymbol{S} \in \mathbb{R}^3 \times \mathbb{R}^3$ opisuje notranji model kamere oz. intrinzične parametre:

$$\boldsymbol{S} = \begin{bmatrix} \alpha_x f & \gamma & c_x \\ 0 & \alpha_y f & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{5}$$

kjer je f goriščna razdalja, α_x in α_y sta skalirna faktorja v horizontalni in vertikalni smeri, (c_x, c_y) je središče slike na optični osi in γ je poševnost oz. koeficient simetrije. Parametre notranjega modela kamere običajno pridobimo v procesu kalibracije kamere.



Slika 32. Perspektivični model kamere

Perspektivični model kamere je nelinearen zaradi izraza ${z_C}^{-1}$, ki pomeni inverz razdalje objekta vzdolž osi z v koordinatnem sistemu kamere C. Čeprav se točke, črte in splošne krivulje pod perspektivično transformacijo ne spreminjajo (npr. točke se transformirajo v točke in črte v črte), daje projecirana slika popačeno sliko realnosti. V splošnem se koti med črtami in razmerja razdalj ne ohranjajo, npr. vzporedne črte se ne transformirajo v vzporedne črte. Model kamere lahko zapišemo tudi z enačbo

$$\underline{\boldsymbol{p}}_P \propto \boldsymbol{S}\boldsymbol{p}_C. \tag{6}$$

Slika objekta nastane na ravnini slike oz. zaslonu za optičnim središčem kamere na razdalji goriščne razdalje f vzdolž negativne z_C polosi in je obrnjena za 180° ter pomanjšana. Ko ustvarjamo vizualno predstavitev modela kamere, lahko predpostavimo, da virtualna neobrnjena slika nastane pred optičnim središčem kamere na pozitivni strani z_C polosi, na isti razdalji od optičnega centra kot pri realni sliki kot je prikazano na sliki 32.

3.1.1 Distorzije in aberacije

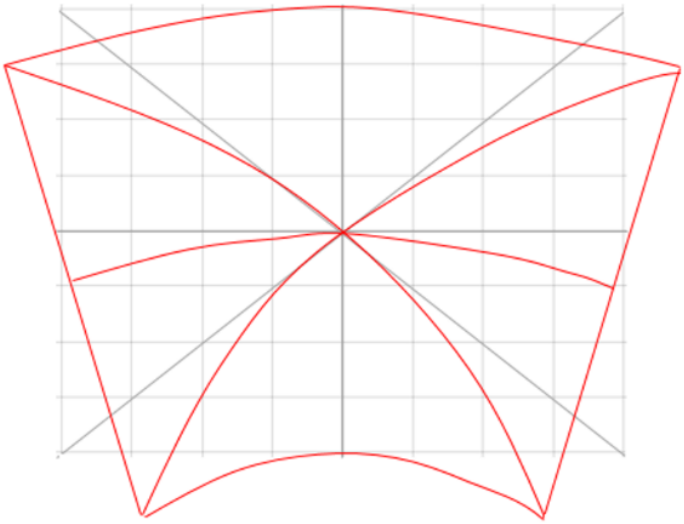
Aberacije slabšajo kakovost slike, nekatere vplivajo tudi na geometrijo. Ločimo naslednje vrste aberacij [49]:

- astigmatizem,
- koma,
- ukrivljenost polja,
- barvne aberacije.

V teh primerih je degradacija kakovosti slike precej vidna, zato se proizvajalci trudijo zmanjšati vpliv degradacije tudi v potrošniški opremi.

Distorzije vplivajo na geometrijo slike [49]. So prisotne, vendar ne precej vidne potrošnikom. V metrologiji pa imajo distorzije močan vpliv na točnost meritev. Posledica distorzij je, da linearni model kamere ne zadostuje v praksi. Poznamo radialne in tangencialne distorzije.

Vzrok tangencialne distorzije je neskladnost elementov optičnega sistema. Posledica je, da so točke na sliki premaknjene v tangencialni smeri (slika 33).



Slika 33. Siva mreža v ozadju je idealna slika, rdeča pa slika s tangencialno distorzijo

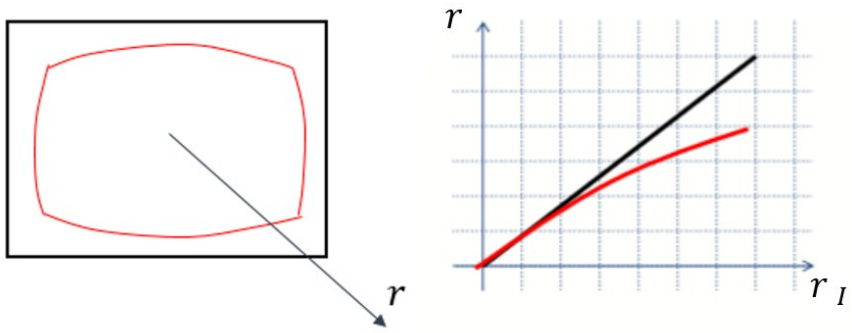
V primeru radialne distorzije so točke na sliki premaknjene v radialni smeri. Poznamo dva tipa radialne distorzije (slika 34):

- »blazinica« (angl. *pincushion*),
- »sodček« (angl. *barrel*), ki je vseprisotna v širokokotnih in potrošniških lečah.

Slika 34. Idealno sliko predstavlja črn pravokotnik, popačeno sliko z radialno distorzijo pa rdeča lika (levo). Primer »blazinice« (sredina) in primer »sodčka« (desno).

Sredinska fotografija: gerogeri [CC BY-NC-ND 2.0] (https://www.flickr.com/photos/tetsu-o/6558660229/)
Desna fotografija: Ats Kurvet [CC BY-SA 4.0] (https://de.wikipedia.org/wiki/Virtual-Reality-Headset)

Efekt radialne distorzije lahko kompenziramo z njenim modeliranjem. Pri oblikovanju modela radialne distorzije v splošnem iščemo nelinearno funkcijo v obliki $r = f(r_I)$ ali $r - r_I = f(r_I)$, kar bi moralo opisati transformacijo radija r_I v r (slika 35), kjer je r_I linearni radij točk slike in r je radij točk slike v prisotnosti radialne distorzije.



Slika 35. Levo: črna slika je idealna slika, rdeča pa z radialno distorzijo. Desno: črn radij je idealni radij točk slike (linearna preslikava), rdeč radij predstavlja prisotnost radialne distorzije.

V enačbi (7) je zapisana polinomska aproksimacija radialne distorzije

$$\Delta r = r - r_I = f(r_I) = k_0 + k_1 r_I + k_2 r_I^2 + k_3 r_I^3 + \cdots + k_n r_I^n, \qquad (7)$$

ki je povzeta s področja fotogrametrije, kjer je distorzija pogosto nepredvidljive narave. Prednost tega zapisa je, da splošni model ne predvideva tipa distorzije. Pomanjkljivost takšnega zapisa je, da pri bistvenih distorzijah potrebujemo veliko koeficientov v zapisu modela. Pri radialnih distorzijah je potreben vsaj zapis drugega reda, da lahko kompenziramo popačenje točk v smeri radija.

Polinomski zapis distorzije je redundanten za čisto radialno distorzijo, ker se koeficienti uporabljajo za prileganje monotono naraščajočim naklonom oz. strminam funkcije $f(r_I)$, namesto za modeliranje nepredvidljivih oblik $f(r_I)$, kot je predvideno v fotogrametriji. Pridobivanje parametrov je optimizacijski proces, ki je lahko nestabilen oz. ne konvergira ali je počasen. Polinomski model je ponekod neustrezen. Pogosto je potrebno malo število parametrov ali inverz.

Obstajajo tudi alternativni modeli, kjer lahko modeliramo določen tip distorzije (monotona distorzija, »sodček«) z manjšim številom koeficientov kot v primeru polinomskega modela, in pa modeli s parametri, povezani s fizično merljivimi lastnosti leč (npr. vidni kot in goriščna razdalja).

Nekaj alternativnih modelov, ki so jih opisali avtorji v člankih [50], [51]:

- FE-transformacija (angl. *Fish-Eye transformation*)
- FOV-model (angl. *Field-Of-View model*)

Pri FE-transformaciji modeliramo radij z enačbo

$$r = s\log(1 + \lambda r_I)\,, \tag{8}$$

kjer je sta s in λ je skalarja, s katerima lahko kontroliramo relativni premik točk na sliki. Pri FOV-modelu pa modeliramo radij z enačbo

$$r = \frac{1}{\omega}\arctan\left(2\, r_I \tan\frac{\omega}{2}\right), \tag{9}$$

kjer je ω vidno polje kamere. Parametre distorzije običajno določimo s postopkom kalibracije kamere.

3.1.2 Epipolarna geometrija

Epipolarna geometrija je poseben primer geometrije več pogledov (angl. *multiview geometry*) [1]. Zaradi lastnosti te geometrije, je le-ta pomembna za rekonstrukcijo scene in različne izvedbe algoritmov strojnega vida (npr. ujemanje točk na slikah in ocenjevanje lege kamere glede na posnete slike).

Recimo, da rotacijska matrika $\boldsymbol{R}_{C_2}^{C_1}$ in translacijski vektor $\boldsymbol{t}_{C_2}^{C_1}$ opisujeta relativno medsebojno lego dveh kamer:

$$\boldsymbol{p}_{C_1} = \boldsymbol{R}_{C_2}^{C_1}\boldsymbol{p}_{C_2} + \boldsymbol{t}_{C_2}^{C_1}. \tag{10}$$

Ob predpostavki, da središči kamer ne sovpada, tj. $\boldsymbol{t}_{C_2}^{C_1} \neq \boldsymbol{0}$, in da imata kameri identične notranje parametre, lahko s kratko matematično manipulacijo pridemo do relacije oz. enačbe (11):

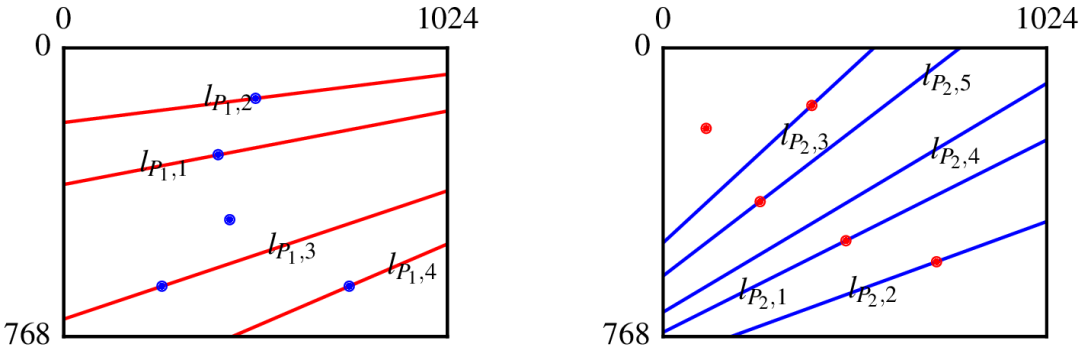
$$\begin{aligned}
\boldsymbol{S}^{-1}\underline{\boldsymbol{p}}_{P_1} &\propto \boldsymbol{R}_{C_2}^{C_1}\boldsymbol{S}^{-1}\underline{\boldsymbol{p}}_{P_2} + \boldsymbol{t}_{C_2}^{C_1},\\
\left[\boldsymbol{t}_{C_2}^{C_1}\right]_\times \boldsymbol{S}^{-1}\underline{\boldsymbol{p}}_{P_1} &\propto \left[\boldsymbol{t}_{C_2}^{C_1}\right]_\times \boldsymbol{R}_{C_2}^{C_1}\boldsymbol{S}^{-1}\underline{\boldsymbol{p}}_{P_2},\\
0 &= \underline{\boldsymbol{p}}_{P_1}{}^{T}\boldsymbol{S}^{-1}\left[\boldsymbol{t}_{C_2}^{C_1}\right]_\times \boldsymbol{R}_{C_2}^{C_1}\boldsymbol{S}^{-1}\underline{\boldsymbol{p}}_{P_2},\\
0 &= \underline{\boldsymbol{p}}_{P_1}{}^{T}\boldsymbol{F}\underline{\boldsymbol{p}}_{P_2}.
\end{aligned} \tag{11}$$

V matematičnem postopku je križni produkt med dvema vektorjema $\boldsymbol{a}^T = [a_1\ a_2\ a_3]$ in $\boldsymbol{b}^T = [b_1\ b_2\ b_3]$ zapisan kot $\boldsymbol{a} \times \boldsymbol{b} = [\boldsymbol{a}]_\times \boldsymbol{b}$, kjer je $[\boldsymbol{a}]_\times$ poševno simetrična matrika

$$[\boldsymbol{a}]_\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}. \tag{12}$$

Matrika $\boldsymbol{F}$ je fundamentalna matrika, ki opisuje epipolarne omejitve v enačbi (11). Točka $\underline{\boldsymbol{p}}_{P_1}$ se nahaja na premici $\boldsymbol{F}\underline{\boldsymbol{p}}_{P_2}$ na prvi sliki in točka $\underline{\boldsymbol{p}}_{P_2}$ leži na premici $\boldsymbol{F}^T\underline{\boldsymbol{p}}_{P_1}$ na drugi sliki (slika 36). Premico $\boldsymbol{F}^T\underline{\boldsymbol{p}}_{P_1}$ lahko označimo z $\boldsymbol{l}_{P_1}$, ki predstavlja epipolarno premico na drugi sliki in

je povezana s točko $\underline{\boldsymbol{p}}_{P_1}$ na prvi sliki (slika 36). Enako lahko označimo premico $\boldsymbol{F}^T\underline{\boldsymbol{p}}_{P_2}$ z $\boldsymbol{l}_{P_2}$, ki predstavlja epipolarno premico na prvi sliki in je povezana s točko $\underline{\boldsymbol{p}}_{P_2}$ na drugi sliki.



Slika 36. Točke in epipolarne premice na dveh slikah

Naslednja pomembna relacija v strojnem vidu je

$$\boldsymbol{p}_{C_1}^T\boldsymbol{E}\boldsymbol{p}_{C_2} = 0, \tag{13}$$

kjer je $\boldsymbol{E}$ znana kot esencialna matrika:

$$\boldsymbol{E} = \left[\boldsymbol{t}_{C_2}^{C_1}\right]_\times \boldsymbol{R}_{C_2}^{C_1}. \tag{14}$$

Relacija med esencialno in fundamentalno matriko je

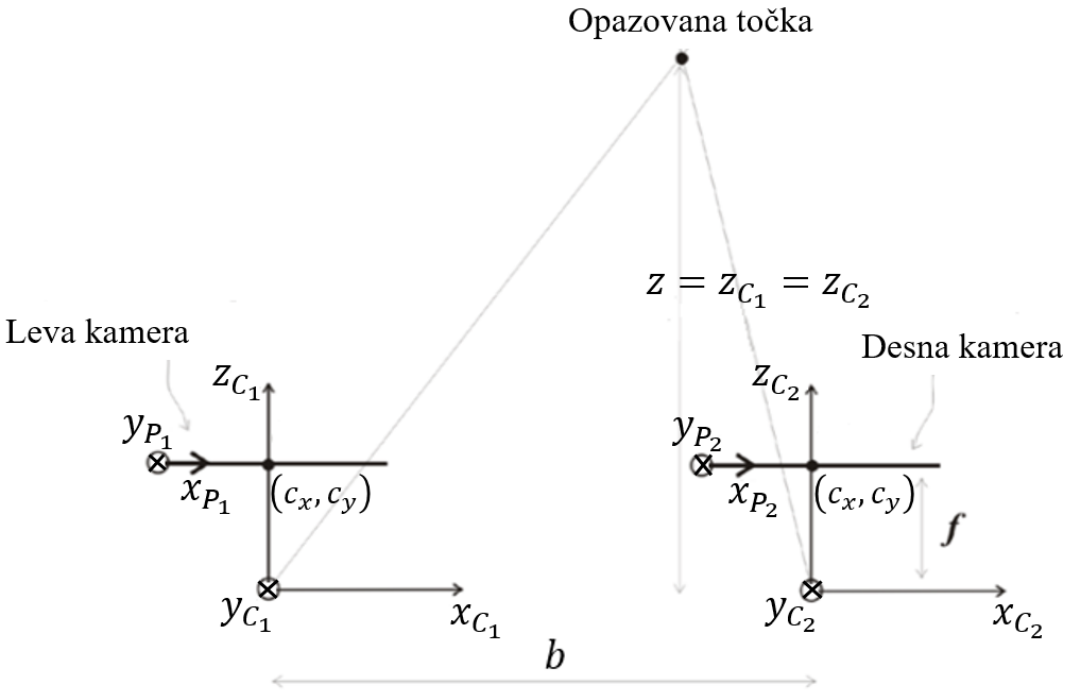
$$\boldsymbol{E} = \boldsymbol{S}^T\boldsymbol{F}\boldsymbol{S}. \tag{15}$$

Epipolarno omejitev lahko uporabimo za izboljšane ujemanja slikovnih točk med več slikami, ki pripadajo isti točki v opazovani tridimenzionalni sceni. Ker mora korespondenčni par točke $\underline{\boldsymbol{p}}_{P_1}$ na prvi sliki ležati na premici $\boldsymbol{F}^T\underline{\boldsymbol{p}}_{P_1}$ na drugi sliki, se dvodimenzionalno iskanje po celotni sliki poenostavi v enodimenzionalno iskanje vzdolž epipolarne premice. Zatorej je proces ujemanja točk znatno pohitren in olajšan ter je lahko bolj robusten, saj so tista ujemanja, ki ne zadovoljujejo epipolarni omejitvi, zavrnjena. Digitalne slike običajno poravnamo tako, da epipolarne premice potekajo po slikovnih vrsticah slik v stereo paru, kar še poenostavi in pohitri proces iskanja parov.

3.1.3 Triangulacija

Da rekonstruiramo informacijo o globini potrebujemo dve sliki istega prizora, kar lahko opišemo z modelom stereo kamere, ki vsebuje dve časovno sinhronizirani kameri. Predpostavimo, da so notranji parametri obeh kamer enaki in da sta optični osi obeh kamer poravnani ter postavljeni na znani medsebojni razdalji b. Na sliki 37 je predstavljen model stereo kamere. Določena točka v prostoru se preslika v slikovno ravnino leve in desne kamere in je podana s koordinatama (x_{P_1}, y_{P_1}) na slikovni ravnini leve kamere in (x_{P_2}, y_{P_2}) na slikovni

ravnini desne kamere. Iz te informacije in poznane postavitve kamer lahko ocenimo položaj opazovane točke v koordinatnem sistemu leve kamere s triangulacijo [1]. Ko izračunamo položaj vseh opazovanih točk, dobimo t. i. oblak točk oz. tridimenzionalno predstavitev opazovanega prizora v kartezičnih koordinatah $(x_{C_1}, y_{C_1}, z_{C_1})$. V koordinatnem sistemu kamere koordinata z_{C_1} predstavlja globino.

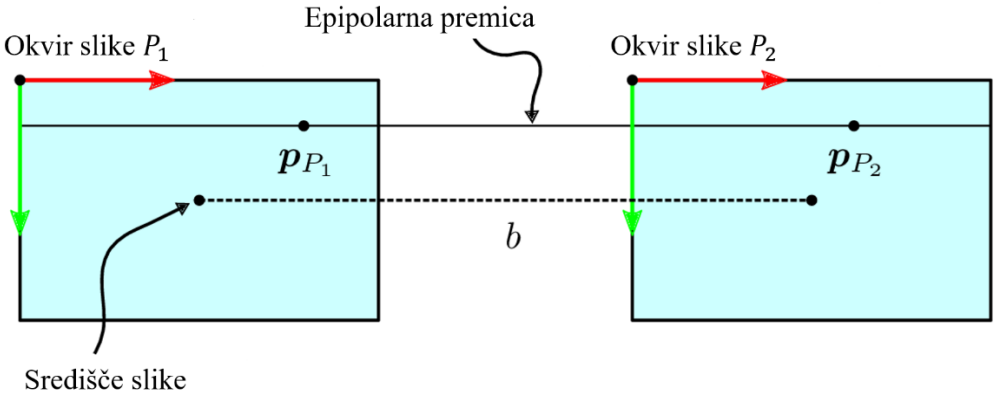


Slika 37. Model stereo kamere

Če so pari na slikah poiskani in če sta modela obeh kamer enaka, lahko globino točke na obeh slikah (z_{C_1} in z_{C_2}) izračunamo z rešitvijo niza enačb (16), pri čemer uporabimo metodo najmanjših kvadratov

$$\boldsymbol{S}^{-1}\underline{\boldsymbol{p}}_{P_1} z_{C_1} - \boldsymbol{R}_{C_2}^{C_1}\boldsymbol{S}^{-1}\underline{\boldsymbol{p}}_{P_2} z_{C_2} = \boldsymbol{t}_{C_2}^{C_1}. \tag{16}$$

Matrika $\boldsymbol{S}$ predstavlja notranji model kamere, $\underline{\boldsymbol{p}}_{P_1}$ in $\underline{\boldsymbol{p}}_{P_2}$ sta točki prve in druge slike, $\boldsymbol{R}_{C_2}^{C_1}$ je rotacijska matrika in $\boldsymbol{t}_{C_2}^{C_1}$ translacijski vektor. Rekonstrukcijski problem se lahko poenostavi v kanonično stereo konfiguracijo dveh kamer, ker so epipolarne premice vzporedne in epipolarna premica točke $\boldsymbol{p}_{P_1}$potuje na prvi sliki skozi točko $\boldsymbol{p}_{P_1}$, kot tudi skozi točko $\boldsymbol{p}_{P_2}$ na drugi sliki. V taki konfiguraciji je slika ene kamere premaknjena vzdolž x-osi (odvisno od postavitve koordinatnega sistema kamere) za vrednost razdalje med kamerama b (slika 38).



Slika 38. Kanonična stereo konfiguracija

V primeru digitalne slike to pomeni, da se pari točk, ki se ujemajo na prvi in drugi sliki, nahajajo v isti vrstici.

Pozicijo točke prizora v koordinatnem sistemu prve kamere lahko izračunamo z enačbo (17), kjer smo zaradi preprostosti predpostavili, da so skalirni faktorji enaki ($\alpha_x = \alpha_y = \alpha$) in faktor poševnosti enak nič ($\gamma = 0$)

$$p_{C_1}^T = \frac{b}{d}\begin{bmatrix} x_{P_1} - c_x & y_{P_1} - c_y & \alpha f \end{bmatrix}. \tag{17}$$

Spremenljivka d je dispariteta in jo računamo kot $d = x_{P_1} - x_{P_2}$. Spremenljivka f predstavlja goriščno razdaljo, spremenljivki c_x in c_y pa središče okvirja slike. Iz zadnjega elementa enačbe (18) je razvidno, da lahko izračunamo globino scene preko disparitete, in sicer preko njenega inverza

$$z_{C_1} = \alpha f b d^{-1}. \tag{18}$$

Za vse točke, ki se nahajajo pred kamero, je dispariteta pozitivna, $d \geq 0$. To lastnost lahko izkoristimo tudi pri iskanju parov.

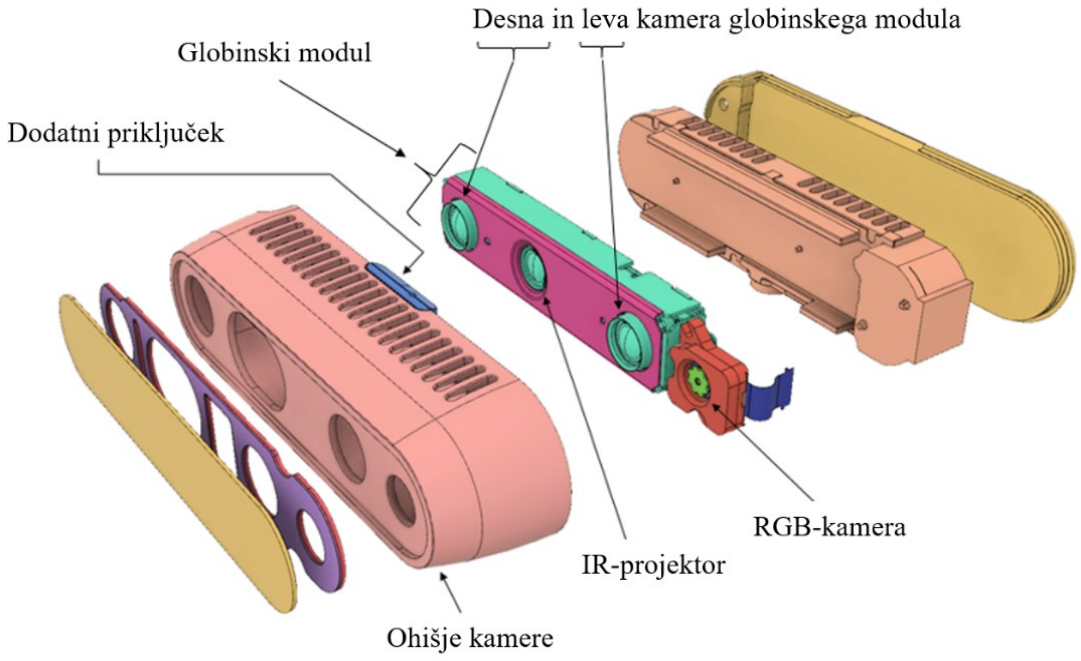
Triangulacijske tehnike so zelo občutljive na popačenja v optičnem sistemu skozi čas [52]. Npr. za strukturirano svetlobo je kritično, da se vzorec, ki ga projeciramo s projektorjem, ne spreminja skozi čas ali s temperaturo, medtem ko so stereo sistemi na to neobčutljivi. Za stereo sisteme in sisteme s strukturirano svetlobo moramo vsake spremembe skozi čas v optičnih notranjih parametrih (lokacija in orientacija leče) in zunanjih parametrih (koti med levo in desno sliko v stereo sistemu ali med sliko in projektorjem v sistemu s strukturirano svetlobo) minimizirati. Tovrstne spremembe vplivajo na kvaliteto in natančnost globine. To se lahko delno reši z izgradnjo miniaturnega optičnega sistema, ki je mehansko izoliran od zunanjega sveta, tako da zunanje sile nanj ne vplivajo.

3.2 Merjenje globine s stereo kamero

Stereo tehnologije za zaznavanje uporabljajo dve kameri za izračun globine in omogočajo napravam da vidijo, razumejo in se učijo iz svojega okolja. Stereo kamere Intel RealSense delujejo tako v zaprtih prostorih kot na prostem v najrazličnejših svetlobnih pogojih. Vgrajeni procesor Intel RealSense Vision Processor D4 izvaja vse izračune globine na kameri. Kamera tako na izhodu vrača globinsko sliko. Na sliki 39 je prikazana globinska kamera D435i.

Slika 39. Globinska kamera Intel Realsense D435i

Stereo kamere serije Intel RealSense so zasnovane za enostavno nastavitev in prenosljivost z visoko ločljivostjo in vključujejo aktivni infrardeči (IR) stereo s standardnim ali širokim vidnim poljem. Na sliki 40 so prikazani sestavni deli globinske kamere D435i, ki je uporabljena v magistrski nalogi. Na sliki 40 je označen tudi dodatni priključek, ki omogoča povezavo več kamer.

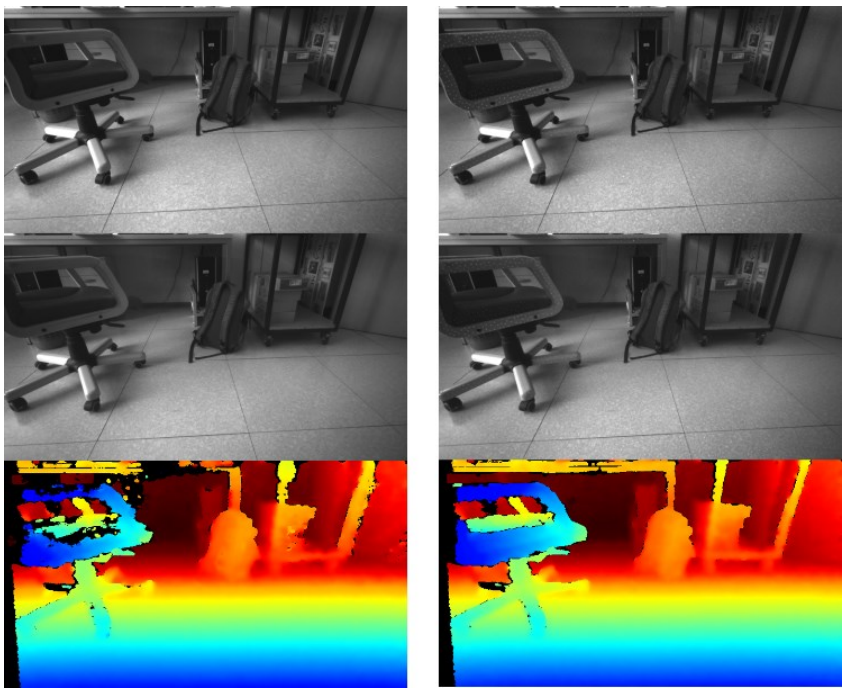


Slika 40. 3D model globinske kamere D435i

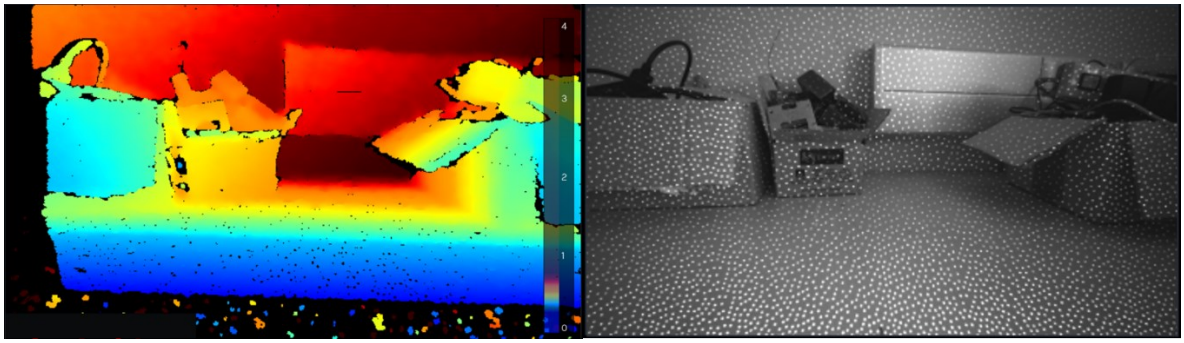
Ena izmed pomembnih omejitev pasivnih stereo sistemov je, da ima težave pri iskanju parov (in s tem tudi globine) na predmetih, ki imajo zelo malo teksture, kot je povsem čista homogena plošča ali tabla [52]. Boljši kot so optični sistemi (kakovost slike) in algoritmi,

boljši so pri prepoznavanju tekstur z uporabo informacije iz več barvnih kanalov. Kljub temu lahko globinski kameri povzročajo težave vogalni primeri oz. vogali. V tem primeru si lahko pomagamo z zunanjim projektorjem za projiciranje snopa svetlobe, ki ustvari teksturo na prizoru, na primer s projiciranjem 100000 majhnih pik na psevdo-naključnih lokacijah. V tej konfiguraciji je sistem znan kot konvencionalni aktivni stereo vid in je po lastnostih zelo podoben sistemom s strukturirano svetlobo. Običajno je za aktivne stereo sisteme izbrana infrardeča svetloba z valovno dolžino, ki je večja kot 850 nm, tako da je ljudem nevidna.

Na sliki 41 je prikazano zajemanje in prikazovanje globinske slike z Intel-ovim programskim orodjem *Intel RealSense Viewer*. Najprej smo na sceno z IR-projektorjem projecirali vzorce v obliki točk, nato pa smo IR-projektor izklopili in primerjali izhodno globinsko sliko v obeh primerih. Opazimo, da s projekcijo vzorcev na sceno izboljšamo proces iskanja parov na slikah stereo para in s tem globinsko sliko. Na sliki 42 je predstavljen nazornejši prikaz projeciranega vzorca točk.



Slika 41. Levo: slika leve kamere (zgoraj) in slika desne kamere (na sredini) ter globinska slika (spodaj) v primeru z izključenim IR-projektorjem; Desno: globinska slika (spodaj) v primeru z vključenim IR-projektorjem.



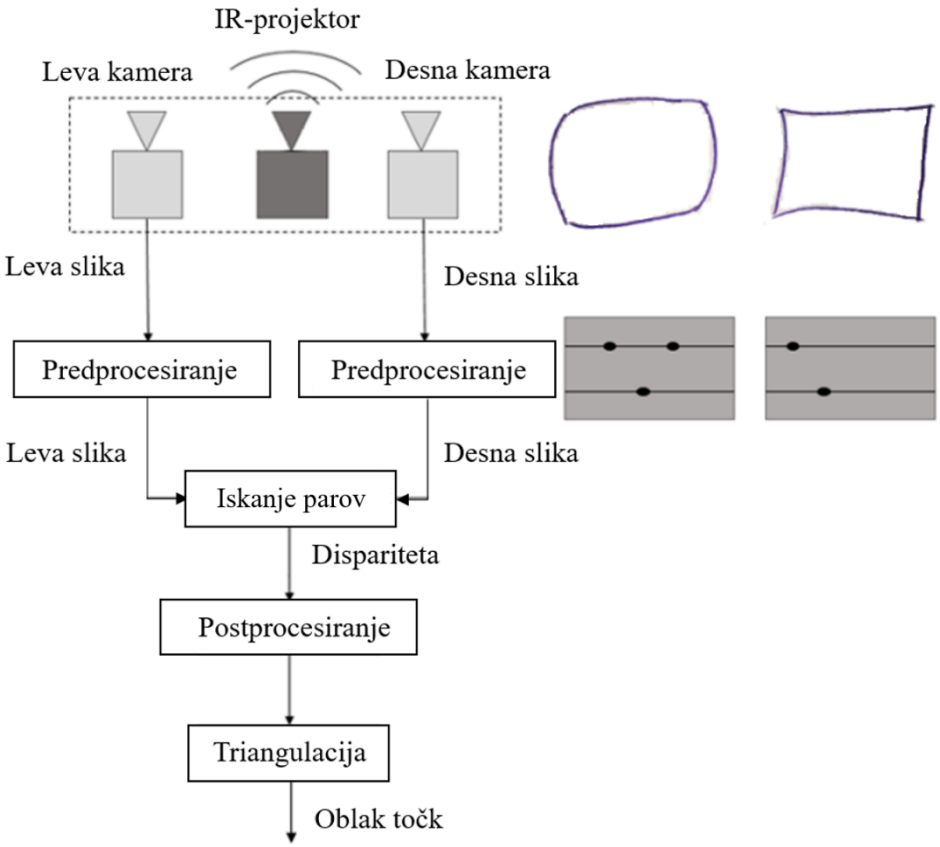
Slika 42. Levo: globinska slika; Desno: IR-projekcija vzorcev na sceno.

Intel RealSense tehnologija podpira širok spekter operacijskih sistemov in programskih jezikov [53]. Intel RealSense SDK 2.0 omogoča, da iz kamere izvlečemo podatke o globini in jih interpretiramo na izbrani platformi – Windows, Linux, MacOS itn. SDK (angl. *Software Development Kit*) je zbirka programske opreme, ki se uporablja za razvoj aplikacij za določeno napravo ali operacijski sistem. SDK ponuja tudi odprtokodne primere kod in vmesnike (angl. *wrappers*) za različne programske jezike in platforme (Python, Node.js, C#, .NET, C in C++). Intel ponuja integracijo s tehnologijami tretjih oseb, kot so ROS, Unity, OpenCV, PCL, Matlab in druga *Intel RealSense* orodja, ki lahko pospešijo razvoj projekta.

Da bi lahko aplikacije izboljšale informacijo o globini v vsaki situaciji, je globinski kameri D435i dodana inercialna merilna enota IMU (angl. *inertial measurement unit*). Enota IMU vsebuje žiroskop in pospeškometer in se uporablja za zaznavanje premikov in rotacije v šestih prostostnih stopnjah [53]. Na ta način lahko zaznamo vrtenje in gibanje v treh oseh ali tako imenovane Eulerjeve kote (φ, θ, ψ), ki opisujejo orientacijo togih teles z rotacijo okoli osi (x, y, z). IMU se uporablja v aplikacijah, kot so igralne in kazalne naprave in tudi za stabilizacijo slike.

Vključevanje IMU-enote v stereo sistem odpira vrata za izvedbo osnovne sočasne lokalizacije in gradnje zemljevida (SLAM) [1] in sledilne aplikacije ter omogoča boljšo poravnavo v oblaku točk.

Na sliki 43 je prikazan diagram poteka, ki prikazuje postopek pridobivanja informacije o globini oz. oblaka točk pri delu z globinsko kamero D435i.



Slika 43. Diagram poteka pridobivanja oblaka točk

V procesu pridobivanja oblaka točk najprej zajamemo sliki z levo in desno kamero. Pri tem lahko na sceno z IR-projektorjem projeciramo svetlobni vzorec, s čimer pripomoremo k boljšemu rezultatu iskanja disparitetne slike. Zajeti sliki sta lahko popačeni in neporavnani. V postopku predprocesiranja lahko s parametri popačenja, ki jih dobimo v postopku kalibracije, zajeti sliki popravimo ter poravnamo. V postopku predprocesiranja lahko izboljšamo tudi kvaliteto slike (npr. kontrast slike). S poravnavo slik omogočimo, da v postopku iskanja parov potekajo epipolarne premice po vrsticah slik, kar pohitri in poenostavi iskanje parov. Objekti, ki so bližje stereo kameri, bodo na epipolarni premici imeli večji relativni premik v horizontalni smeri [52]. Ta relativni premik je znan kot dispariteta in ga izračunamo za vsako točko na sliki. Iskanje ujemanja takšnih točk je na področju strojnega vida znano kot reševanje problema iskanja parov. Ko v postopku iskanja parov najdemo pare točk, izračunamo disparitetno sliko, ki jo lahko v postopku postprocesiranja dodatno obdelamo. Algoritmi postprocesiranja so predstavljeni v podpoglavju 3.2.6. Iz disparitetne slike izračunamo globinsko sliko oz. oblak točk s postopkom triangulacije (poglavje 3.1.3).

3.2.1 Kalibracija kamere

Stereo kalibracija je postopek izračuna parametrov stereo kamere [40]. S pomočjo kalibracije lahko pridobimo rotacijsko matriko in translacijski vektor, ki določata geometrijsko relacijo med dvema kamerama, ter parametre modela distorzije leče, s katerim lahko popravimo popačenja, ki so posledica nepopolne leče ali zgradbe optičnega sistema. V nadaljevanju je opisana dinamična kalibracija, ki so jo opisali avtorji v [54].

Globinska kamera Realsense D435i je tovarniško kalibrirana, vendar pa ima uporabnik možnost ponovne kalibracije. Pri močnih udarcih ali padcih kamere lahko na ta način kalibriramo kamero in odpravimo težave z zajemanjem podatkov o globini. Pri procesu kalibracije ocenjujemo notranje in zunanje parametre [54], ki opisujejo osnovne karakteristike kamere.

Notranji parametri vključujejo:

- goriščno razdaljo, ki je podana v slikovnih elementih kot $[f_x\ f_y]^T$ za levo, desno in RGB kamero;
- točko optičnega središča, ki je podana v slikovnih elementih kot $[c_x\ c_y]^T$ za levo, desno in RGB kamero;
- distorzijo, ki je pogosto podana kot Brown-ov model $[k_1\ k_2\ p_1\ p_2\ k_3]^T$ [55] za levo, desno in RGB kamero.

Zunanji parametri vključujejo:

- rotacijo iz koordinatnega sistema desne kamere v koordinatni sistem leve kamere, ki je določena kot rotacijska matrika R_D^L z dimenzijami 3×3;
- translacijo iz koordinatnega sistema desne kamere v koordinatni sistem leve kamere, ki je določena kot vektor t_D^L z dimenzijami 3×1 podan v milimetrih;
- rotacijo iz koordinatnega sistema RGB-kamere v koordinatni sistem leve kamere, ki je določena kot rotacijska matrika R_{RGB}^L z dimenzijami 3×3;
- translacijo iz koordinatnega sistema RGB-kamere v koordinatni sistem leve kamere, ki je določena kot vektor t_{RGB}^L z dimenzijami 3×1 podan v milimetrih.

Leva kamera je referenčna in njen koordinatni sistem se nahaja v globalnem izhodišču. RGB-parametri veljajo le za globinske module oz. kamere z RGB-senzorjem, ki zaznava barve (npr. globinske kamere D415, D435 in D435i).

Uporabniki kamer lahko uporabljajo svoje algoritme za kalibracijo, vendar pa Intel ponuja že obstoječe rešitve kalibracije. Uporabljajo več vrst t. i. dinamične kalibracije.

Dinamična kalibracija optimizira le zunanje parametre kamere (translacija in rotacija med levo in desno kamero). Optimizacija teh parametrov se izvaja v prostoru, ki ga je postavil uporabnik

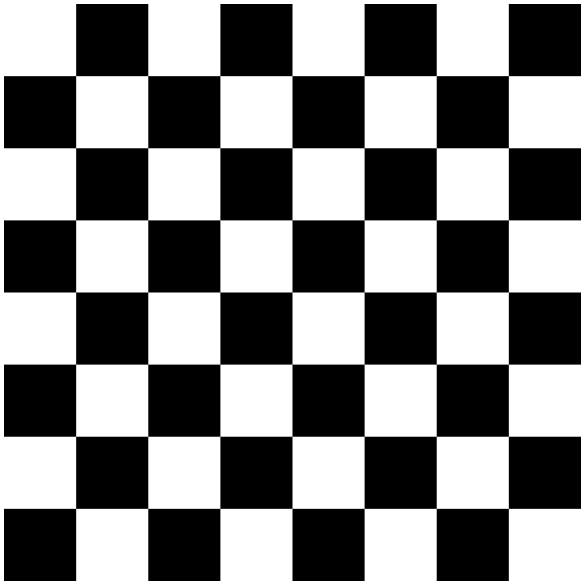
in pri tem malo ali nič ne posreduje v postopku kalibracije. Notranjih parametrov, kot so distorzija, vidno polje in optično središče, ne kalibriramo dinamično. Dinamična kalibracija se izvaja pod predpostavko, da gre za ponovno kalibracijo globinskih modulov oz. kamer po tovarniški kalibraciji ali če so že znani nominalni parametri.

Podprta sta dva različna tipa algoritmov dinamičnih kalibracij:

- Poravnava (angl. *rectification calibration*), ki pomeni poravnavo epipolarnih črt, s čimer pohitrimo in izboljšamo iskanje parov ter zmanjšamo število lukenj v globinski sliki.
- Umerjanje globine (angl. *depth scale calibration*), ki pomeni poravnavo globinske slike zaradi sprememb v položaju optičnih elementov.

Orodje za dinamično kalibracijo podpira te algoritme v dveh načinih delovanja: način s tarčo in brez tarče. Tarčna kalibracija je priporočen pristop, ker vključuje poravnavo kot tudi umerjanje globine in bo dala natančnejše ter doslednejše rezultate.

V načinu s tarčo potrebujemo tarčo, ki je vnaprej določena in jo lahko natisnemo na papir primerne velikosti ali pa jo prikažemo na pametnem telefonu preko aplikacije *Intel RealSense Dynamic Target Tool phone app*. Primer tarče je prikazan na sliki 44.



Slika 44. Tarča za dinamično kalibracijo

Fotografija: Font Awesome Free 5.2.0 by @fontawesome - https://fontawesome.com, [CC BY 4.0] (https://en.wikipedia.org/wiki/File:Font_Awesome_5_solid_chess-board.svg)

Aplikacija v postopku kalibracije pridobi slike kalibracijske tarče in jih pošilja v algoritem za dinamično kalibracijo. Postopek je razdeljen na dva zaporedna koraka: fazo poravnave in fazo skaliranja. Proces zahteva, da zajemamo sočasne slike tarče z več globin. Resolucija popravljenih slik je v seriji globinskih modulov *Intel RealSense D400* določena s formatom slike in je odvisna od modela globinske kamere. Resolucija popravljene slike (leve in desne

kamere) za globinsko kamero D435i je 1280×800. Kalibriramo lahko tudi RGB-kameri, kjer je resolucija popravljene slike 1920×1080.

Proces kalibracije je opisan v naslednjih korakih:

1. Zajamemo sliki leve in desne kamere.
2. Zaznamo kalibracijsko tarčo na obeh slikah.
3. Uporabnik premika natisnjeno tarčo ali telefon, tako da pokrijemo celotno vidno območje (ponavljanje prvega in drugega koraka).
4. Po 15-ih zajetih slikah je proces končan.
5. Sledi preverba napake poravnave in primerjava velikosti izmerjenega vzorca z dejansko vrednostjo.

Slika 45 prikazuje kalibrirani sivinski sliki, zajeti z globinskim modulom, ki sta rezultat faze predprocesiranja, kot smo to prikazali na diagramu poteka (slika 43). Kalibrirani sliki sta poravnani in pripravljeni za postopek iskanja parov, ki je opisan v podpoglavju 3.2.2.

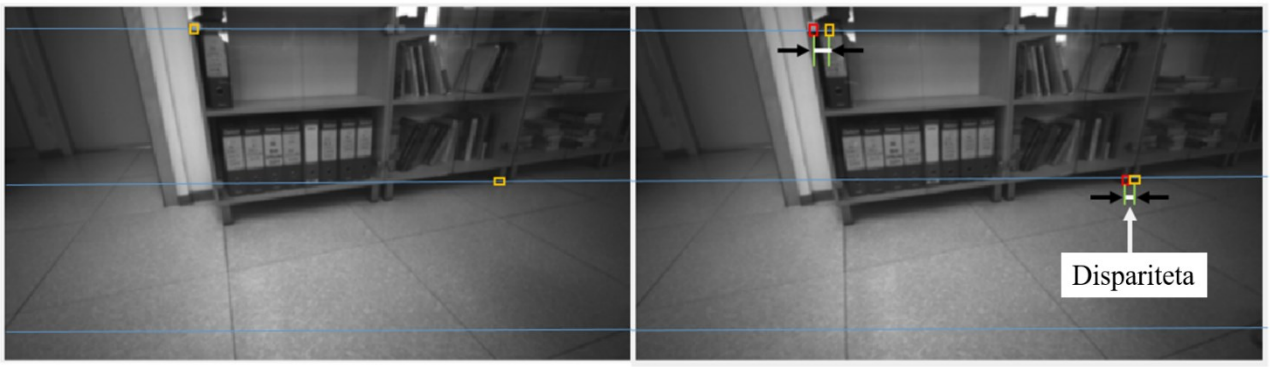
Slika 45. Slika leve kamere (levo) in slika desne kamere (desno)

3.2.2 Iskanje parov

Za točko v levi sliki je potrebno poiskati ustrezno točko v desni sliki. Ko najdemo pare, ki se ujemajo, izračunamo dispariteto. Pri procesu iskanja parov moramo izbrati strategijo iskanja parov in mero podobnosti [56]. Poznamo več strategij iskanja parov, od katerih bomo predstavili korelacijsko metodo (angl. *correlation-based strategy*) in metodo iskanja značilk (angl. *feature-based strategy*).

3.2.3 Korelacijska metoda

Pri korelacijski metodi [56] moramo najprej izbrati velikost področja ali okenca na prvi sliki, s katerim iščemo ujemanje na drugi sliki. Nato določimo območje iskanja, tj. kako daleč iskati točko para. Točke, ki so bolj oddaljene od globinske kamere, bodo na kalibriranih slikah stereo kamere manj narazen. Torej bo območje iskanja vzdolž epipolarne črte manjše oz. krajše (slika 46). Podobno sklepamo, da bodo točke, ki so bližje senzorju, bolj razmaknjene na slikah leve in desne kamere. To pomeni, da je območje iskanja vzdolž epipolarne črte v tem primeru večje oz. daljše.



Slika 46. Epipolarne črte kalibriranih slik in vizualizacija disparitete

Obstaja več mer podobnosti:

- vsota absolutnih razlik (angl. *Sum of Absolute Diferrences – SAD*):

$$SAD(x,y) = \sum_{i,j} |I_l(i,j) - I_r(i+x, j+y)| \,, \tag{19}$$

- vsota kvadratov razlik (angl. *Sum of Square Diferrences – SSD*):

$$SSD(x,y) = \sum_{i,j} \big(I_l(i,j) - I_r(i+x, j+y)\big)^2 \,, \tag{20}$$

- križna korelacija (angl. *Cross-Correlation – CC*):

$$CC(x,y) = \sum_{i,j} I_l(i,j) I_r(i+x, j+y) \,, \tag{21}$$

- normalizirana križna korelacija (angl. *Normalized Cross-Correlation – NCC*):

$$NCC(x,y) = \frac{\sum_{i,j} I_l(i,j) I_r(i+x, j+y)}{\sqrt{\sum_{i,j} I_l(i,j)^2} \sqrt{\sum_{i,j} I_r\,(i+x, j+y)^2}} \,, \tag{22}$$

- in druge.

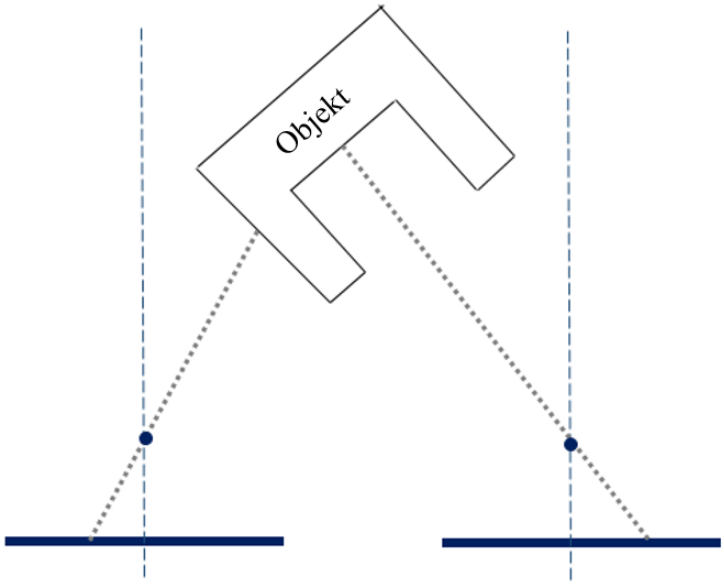
Iščemo najboljše ujemanje točk z izbrano mero podobnosti, ki jo izračunamo kot funkcijo relativnega premika točke $d = (u, v)$, ki je v primeru križne korelacije:

$$S\big(d(i,j)\big) = \sum_{k=-w}^{k=+w} \sum_{l=-w}^{l=+w} I_l(i+k, j+l)\, I_r(i+k+u, j+l+v), \tag{23}$$

kjer je I_l intenziteta slikovnega elementa na levi sliki, I_r pa na desni sliki. Parameter w predstavlja širino in višino izbranega okenca. Z enačbo (23) iščemo relativne premike pri vsaki točki na sliki, ki je podana s koordinatami i in j. Z enačbo (24) pa izberemo ujemajočo se točko s koordinatama i in j, kjer je mera podobnosti največja pri danem relativnem premiku.

$$d(i,j) = argmax_d\{S(d)\} \tag{24}$$

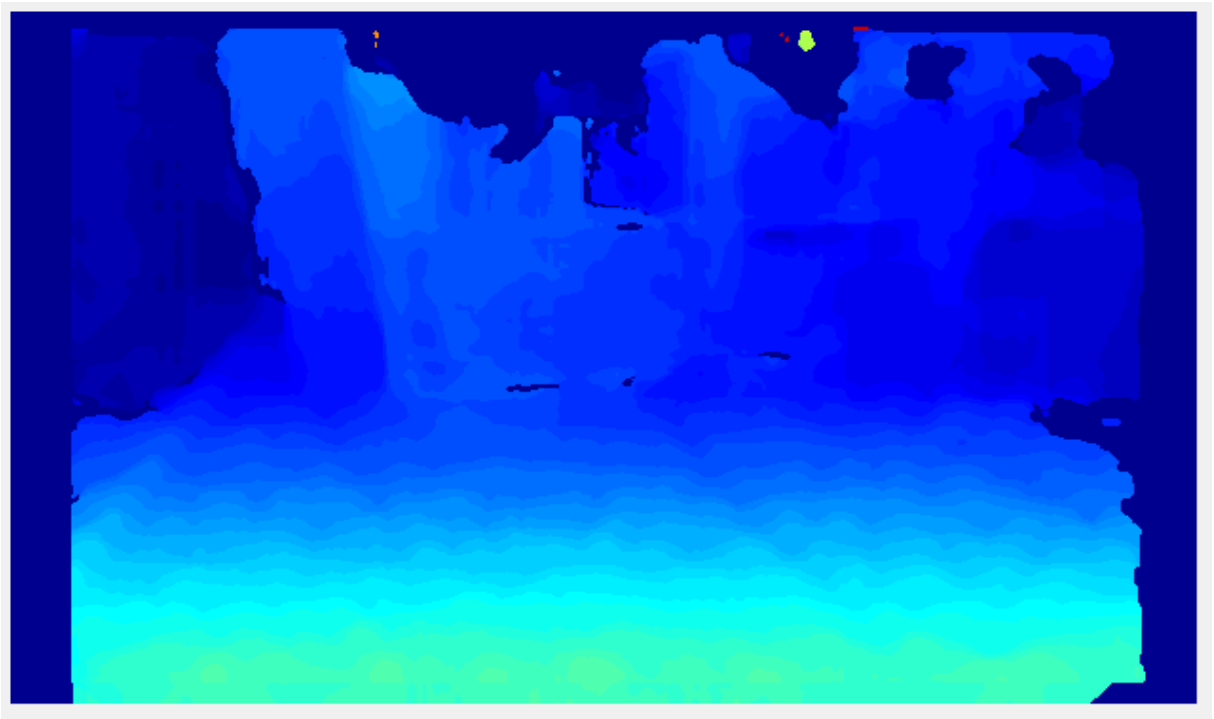
Na območjih slik, ki so brez tekstur (homogena področja), ne moremo iskati parov točk, saj je postopek nezanesljiv. To lahko rešimo tako, da uporabimo stereo vid s strukturirano svetlobo. S tem ne moremo rešiti problema skritih območij (območij, ki niso vidna z obema kamerama), ki lahko vodijo v napačna ujemanja kot posledica nejasnosti oz. dvoumnosti (slika 47).



Slika 47. Skrita območja

Končni rezultat korelacijske metode je gosta disparitetna slika, ki vsebuje gosto informacijo o globini. Slabost te metode je, da lahko proizvede veliko napačnih ujemanj. Za izboljšanje delovanja se zato uporabljajo dodatni mehanizmi, ki omogočajo izbiro pravih parov in izboljšajo disparitetno sliko.

Iz leve in desne slike (slika 46), ki smo ju posneli z globinsko kamero D435i, smo izračunali disparitetno sliko (slika 48) tudi s pomočjo programskega orodja Matlab. To smo storili z uporabo funkcije *disparity* [57]. Oceno disparitetne slike smo pridobili z algoritmom semi-globalnega ujemanja (angl. *semi-global matching)* [58]. Metoda semi-globalnega ujemanja dodatno vsiljuje podobne disparitete v sosednjih blokih oz. okencih. Ta dodatna omejitev omogoča boljšo oceno disparitete.



Slika 48. Disparitetna slika dobljena z Matlabovo funkcijo disparity

Algoritem se izvaja v naslednjih korakih:

1. Z uporabo filtra *Sobel* se izračuna mera kontrasta slike.
2. Izračuna se dispariteta za vsak slikovni element v eni sliki

Za izboljšanje iskanja parov se v tem algoritmu uporabljajo različni dodatni mehanizmi. Pri iskanju parov lahko določimo območje, s katerim omejimo območje iskanja parov vzdolž epipolarnih premic. Tako lahko pare, ki so najdeni izven določenega območja, zavržemo. Za iskanje pravih parov lahko uporabimo dodatni mehanizem, s katerim iščemo za točko na prvi sliki, ujemajočo se točko na drugi sliki. Nato poskusimo za odkrito točko na drugi sliki ponovno poiskati par še na prvi sliki. Če smo v postopku iskanja parov na dveh slikah odkrili različne pare, potem le-te zavržemo.

Poleg omenjenih mehanizmov za izboljšanje iskanja parov lahko upoštevamo tudi teksturo slikovnih elementov. Izračunana dispariteta za slikovne elemente, katerih vrednost teksture pade pod vrednost določene s pragom teksture, zavržemo.

V primeru paralelnih kamer se dispariteta manjša, ko razdalja narašča. Stereo vid deluje dobro za male razdalje, saj napaka narašča s kvadratom razdalje [56]. Z naraščanjem razdalje med levo in desno kamero, se napaka zmanjšuje, vendar pa so slike s stališča perspektive vse bolj različne in posledično se bodo pojavila napačna ujemanja pri iskanju parov. Obstajajo rešitve za ta problem, npr. dodajanje več kamer v stereo sistem, ki imajo različno medsebojno razdaljo (angl. *multi-baseline stereo*).

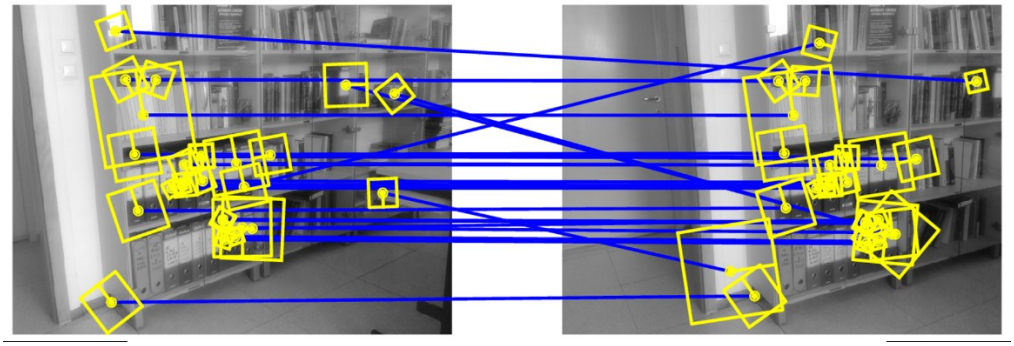
3.2.4 Metoda iskanja značilk

Pri metodi iskanja značilk [1] gre za odkrivanje izrazitih točk na obeh slikah, kot so robovi, vogali, premice itn., ki jih opišemo z značilkami. Značilke definiramo na podlagi lokalne okolice določene izrazite točke tako, da so le-te čim bolj invariantne na različne geometrijske in kvalitativne transformacije. Za vsako točko želimo najti ujemanja teh značilk na levi in desni sliki, pri čemer računamo mero podobnosti med značilkami. Končni rezultat metode iskanja značilk je redka globinska slika. Gosto globinsko sliko lahko v tem primeru dobimo z interpolacijo podatkov.

Prednost te metode v primerjavi s korelacijsko je, da izračuna manj napačnih ujemanj in je manj občutljiva na fotometrične spremembe.

Pomembni algoritmi pri odkrivanju lokalnih značilk so SIFT (angl. *Scale Invariant Feature*) [59], SURF (angl. *Speeded-Up Robust Features*) [60], MSER (angl. *Maximally Stable Extremal Regions*) [61], FAST (angl. *Features from Accelerated Segment Test*) [62] in AGAST (angl. *Adaptive and Generic Accelerated Segment Test*) [63]. Večina teh algoritmov je vključenih v odprtokodno knjižnico za računalniški vid *OpenCV* [41, 42].

Na sliki 49 so s kvadratki označena območja in predstavljajo lokalni vzorec okoli značilke. Ta območja nam pomagajo določiti primerne deskriptorje, ki so običajno predstavljeni kot vektor značilk. Lokalni deskriptorji morajo biti izraziti, zato da omogočimo natančno identifikacijo značilk ne glede na različne spremembe v okolju (npr. spremembe osvetlitve).



Slika 49. Odkrite značilke na dveh slikah istega prizora

Pare značilk lahko poiščemo s primerjanjem razdalj med vektorji značilk, tj. deskriptorji. Glede na tip deskriptorja lahko izberemo različne mere razdalje. Običajno izberemo Evklidsko ali Manhattan razdaljo za deskriptorje z realnimi vrednostmi in Hamming-ovo za delo z binarnimi deskriptorji.

3.2.5 Izračun globine

Z uporabo triangulacije [65] je možno povezati dispariteto vsakega slikovnega elementa z razdaljo. Razdaljo oz. globino računamo z enačbo (18). Resolucija globine η oz. najmanjši korak vrednosti globine se lahko razlikuje glede na dispariteto in jo izračunamo z enačbo (24).

$$\eta = \frac{z_{C_1}^2 \delta}{fb}, \tag{25}$$

kjer je δ korak disparitete in je podan v slikovnih elementih. Pri izračunu resolucije je ključen najmanjši korak disparitete [65]. Algoritmi za izračun stereo globine se običajno vrednotijo glede na korak disparitete, ki jo lahko dosežejo. Algoritmi, ki se izvajajo na procesorjih v globinskih kamerah *Intel Realsense D4xx*, imajo lahko korak disparitete do $\frac{1}{32}$ slikovnega elementa. Da dosežemo tako majhne korake disparitete, je potrebno vključiti množico kompleksnih slikovnih operacij, ki so predmet mnogih raziskav.

Obstajajo omejitve glede tega, kako blizu kamere so lahko objekti in to razdaljo imenujemo najmanjša razdalja Z_{min}. Če se objekti nahajajo na razdalji od kamere, ki je manjša od Z_{min}, potem ne moremo najti parov točk, ker objekt ni v vidnem polju obeh kamer.

Omenimo naj, da se pri uporabljeni kameri (D435i), ne moremo približati bližje kot je to določeno z najmanjšo razdaljo Z_{min}, ki je podana z enačbo

$$Z_{min} = \frac{fb}{126}. \tag{26}$$

V našem primeru je $Z_{min} = 120\ mm$ pri resoluciji 480×270. Najmanjšo razdaljo lahko vedno zmanjšamo z manjšanjem resolucije kamere. V tabeli 3 so prikazane najmanjše razdalje za različne globinske kamere proizvajalca *Intel*, ki jih lahko še uspešno izračunamo glede na izbrano resolucijo kamere [66].

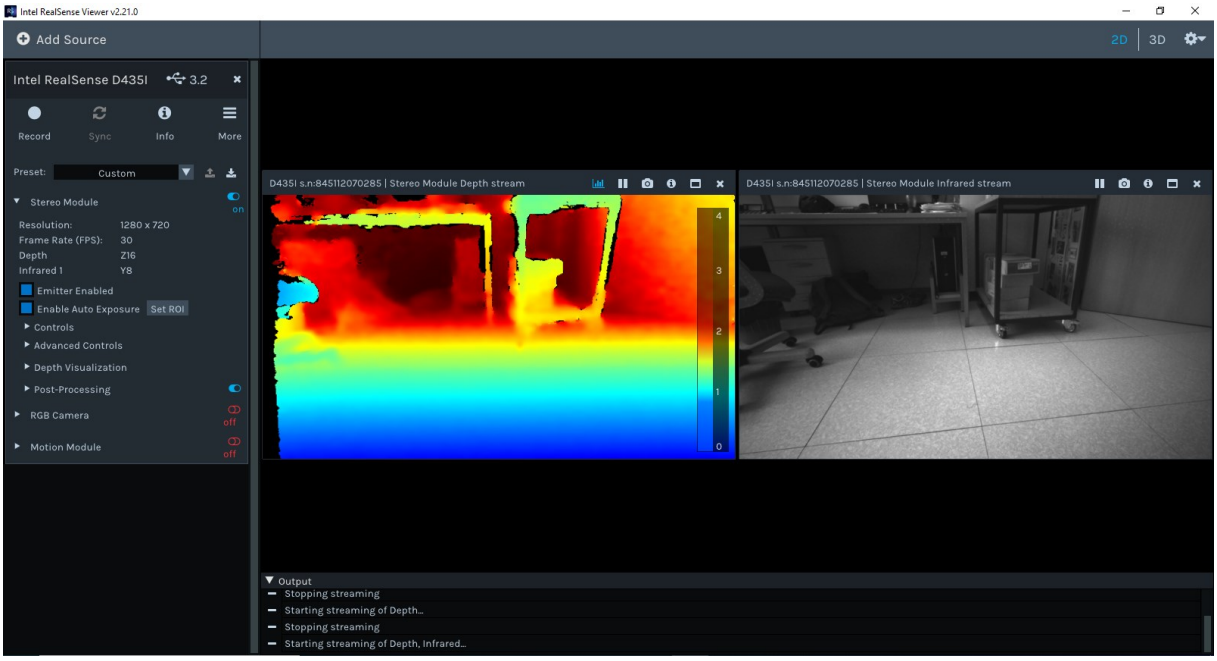
Tabela 3. Najmanjše merljive razdalje globinskih kamer glede na resolucijo slike.

Resolucija	**D400/D410/D415**	**D420/D430**
	Z_{min} [mm]	Z_{min} [mm]
1280 x 720	450	280
848 x 480	310	195
640 x 480	310	175
640 x 360	240	150
480 x 270	180	120
424 x 240	160	105

3.2.6 Postprocesiranje

Globinske kamere *Intel RealSense serije D4xx* lahko prenašajo globino v stvarnem času (tj. podatke o globini) in barvo prizora vse do 90 slik na sekundo [67]. Vso potrebno procesiranje podatkov se izvaja na vgrajenem procesorju D4 ASIC, kar bi moralo pustiti skoraj nič bremena na procesorju uporabnika. Tako se lahko uporabnik osredotoči le na uporabo globinskih podatkov za ustvarjanje različnih aplikacij. Čeprav je mogoče prilagoditi več kot 40 parametrov, ki vplivajo na izračun globine, je treba upoštevati, da ASIC ne opravi nobene naknadne obdelave za čiščenje globine, saj je to prepuščeno aplikacijam višje ravni, če je to potrebno. V tem poglavju se bomo osredotočili na preproste korake postprocesiranja, s katerimi lahko izboljšamo informacijo o globini za določene aplikacije.

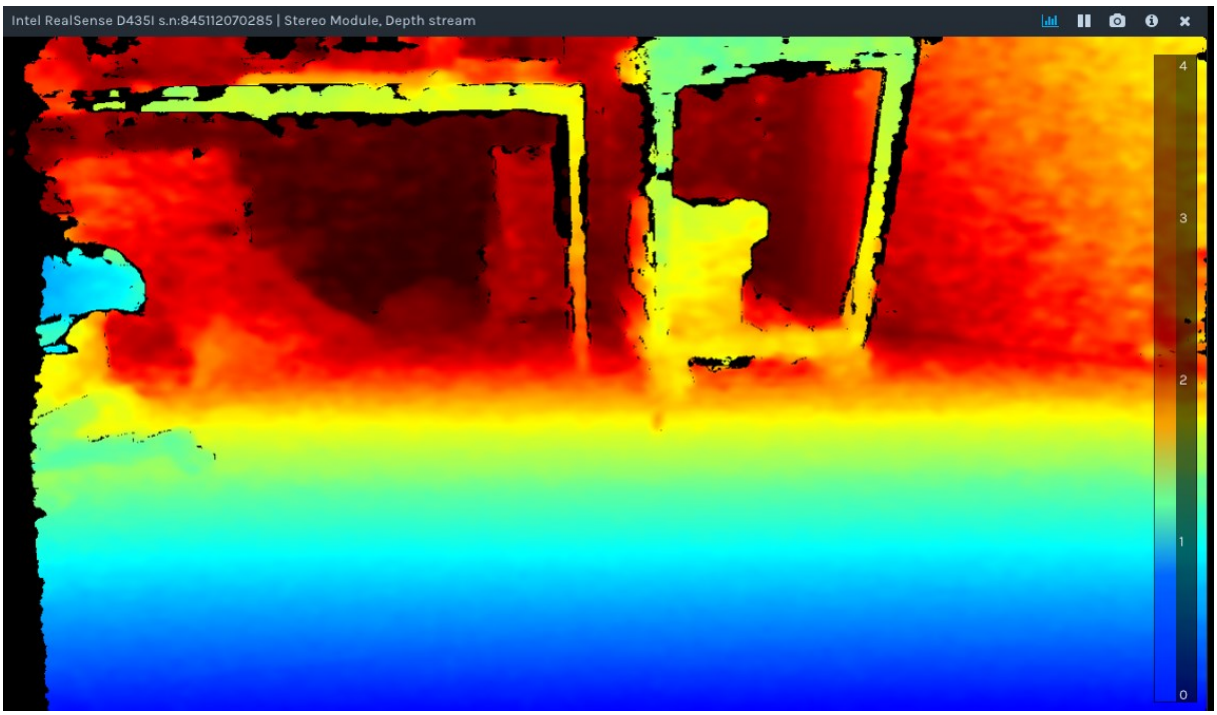
Funkcije postprocesiranja so vključene tudi v aplikacijo *Intel RealSense Viewer* (slika 50), tako da lahko na hiter način raziščemo učinke postprocesiranja. Slika 51 podrobneje prikazuje parametre, ki jih lahko nastavimo pri določenih funkcijah postprocesiranja.



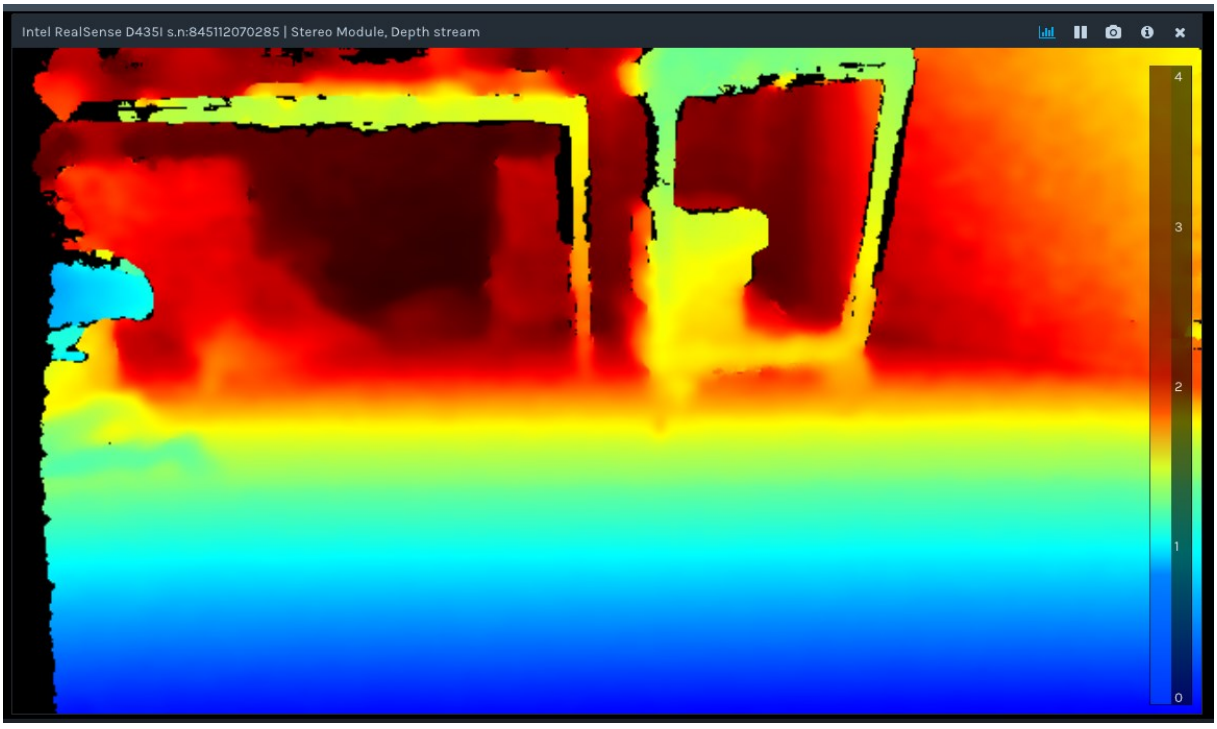
Slika 50. Uporabniški vmesnik za upravljanje s funkcijami postprocesiranja v Intel RealSense Viewer in zajem globinske slike

Slika 51. Podrobnejši pregled uporabniškega vmesnika za upravljanje s funkcijami postprocesiranja

Na sliki 52 je prikazana globinska slika brez izvedbe postopka predprocesiranja. Na sliki 53 lahko opazimo učinke uporabe funkcij postprocesiranja.



Slika 52. Globinska slika brez postprocesiranja



Slika 53. Globinska slika s postprocesiranjem

Funkcije postprocesiranja so:

- Filter decimacije (angl. *decimation filter*), ki zmanjša resolucijo globinske slike.
- Prostorski filter (angl. *spatial filter*), ki uporablja metodo ohranjanja robov, medtem ko gladi globinske podatke.
- Časovni filter (angl. *temporal filter*), ki filtrira globinske podatke z upoštevanjem prejšnjih globinskih slik.
- Filter lukenj (angl. *hole filling filter*)

V nadaljevanju bomo opisali metode postprocesiranja, ki smo jih povzeli po prispevku [68].

Decimacija pomeni, da preprosto zmanjšamo globinsko sliko tako, da vzamemo npr. vsaki n-ti slikovni element. Priporočeno je bolj inteligentno podvzorčenje, in sicer uporaba metode neničelnega povprečja (angl. *non-zero mean*) ali neničelne mediane (angl. *non-zero median*) za slikovni element in njegove sosede. Ker vsi stereo algoritmi vključujejo konvolucijske operacije, je zmanjšanje resolucije surove slike koristno za zmanjšanje računskega časa za aplikacije na višjem nivoju. Faktor dve reduciranja ločljivosti bo pospešil naknadno obdelavo oz. računski čas do 4-krat, faktor 4 pa vse do 16-krat. Ko je globinska karta stisnjena na manjšo ločljivost, lahko uporabimo bolj kompleksne prostorske in časovne filtre. Priporočeno je, da se najprej uporabi prostorski filter z ohranjanjem robov.

Prostorski filter bo zgladil šum globine in ohranil robove, medtem ko bo površine poskušal narediti bolj ploščate. Vendar je treba paziti, da izberemo takšne parametre, ki ne bodo preveč agresivno odstranjevali značilnosti slike. Priporočena je uporaba prostorskega filtra v domeni disparitete in eksperimentiranje s postopnim povečevanjem praga, dokler ne dobimo ustreznih rezultatov glede na namen uporabe.

Časovni filter deluje tako, da povprečimo globinske slike skozi čas, kadar je to možno. Pri tem pazimo, da izključimo točke, kjer so luknje oz. kjer je globina enaka nič. Ker je v globinskih podatkih prisoten šum, ki se kopiči skozi čas, je priporočljivo da se uporabi digitalen IIR-filter (angl. *Infinite Impulse Response*) [69], ki ima neskončni impulzni odziv.

Nekatere aplikacije ne prenašajo lukenj v globini. Na primer, za področje fotografije z napredno globino (angl. *depth-enhanced photography*) je pomembno, da obstajajo vrednosti globine za vsak slikovni element, četudi gre za ugibanje. Za to je potrebno luknje zapolniti z najboljšimi ugibanji na podlagi sosednjih vrednosti ali RGB-slike.

Implementirane so tri metode zapolnjevanje lukenj (slika 54):

1. Luknjo oz. slikovni element zapolnimo z veljavno vrednostjo levega slikovnega elementa.

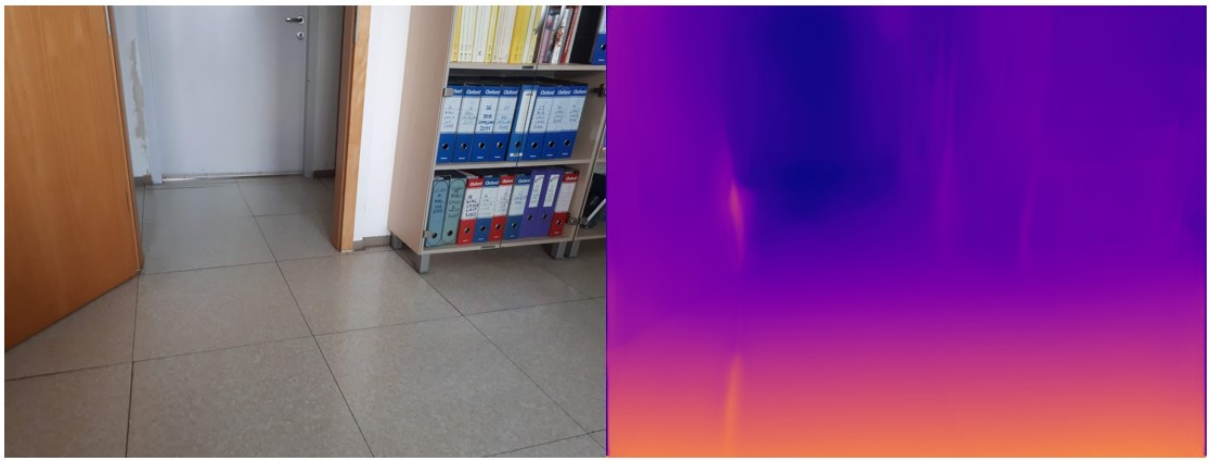
2. Luknjo oz. slikovni element zapolnimo z največjo (najbolj oddaljeno) med veljavnimi petimi zgornjimi levimi in spodnjimi vrednostmi slikovnih elementov (uporabimo pri globinski sliki).
3. Luknjo oz. slikovni element zapolnimo z najmanjšo med veljavnimi petimi vrednostmi zgornjih levih in spodnjih slikovnih elementov (uporabimo pri disparitetni sliki).

Slika 54. Uporaba algoritma polnjenja lukenj z uporabo bloka za obdelavo diskretnih lukenj ima tri možnosti za polnjenje.

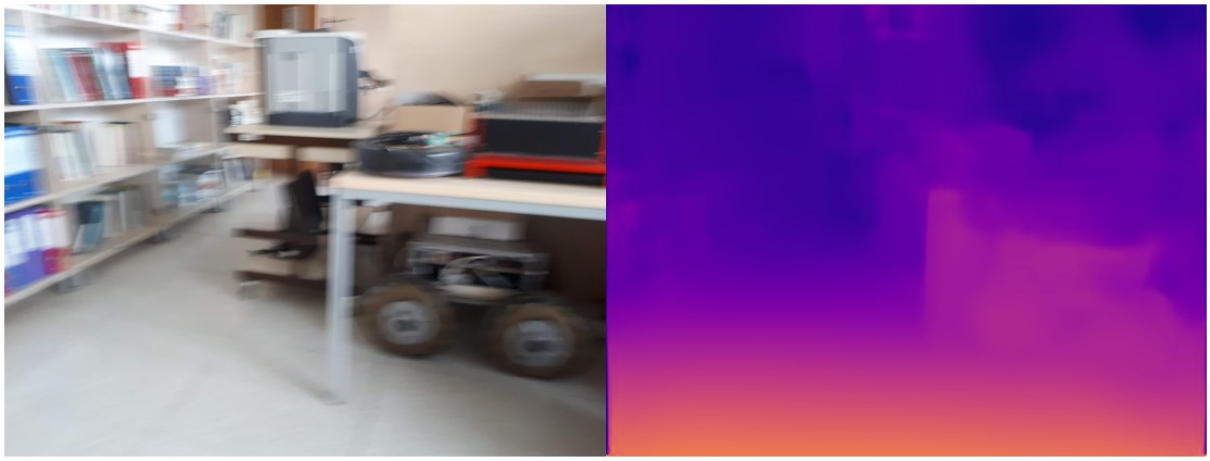
3.2.7 Alternativni pristopi

Slike vsebujejo mnogo informacij oz. dodatne informacije, na podlagi katerih lahko izboljšamo zaznavanje globine (npr. sence, intenziteta svetlobe, linije, robovi, koti, prekrivanje objektov itn.). Z upoštevanjem dodatne informacije bi lahko prišli do bolj natančne ocene globine, vendar so metode, ki obravnavajo te informacije običajno kompleksne in časovno potratne.

Na slikah 55 in 56 sta prikazani oceni globine le na podlagi ene slike. Globinski sliki sta rezultat metode »*monodepth*«, s katero izvajamo nenadzorovano predvidevanje globine posamične slike z uporabo konvolucijske nevronske mreže [70]. Nevronska mreža je bila naučena na množici slik notranjih prostorov z znano disparitetno sliko. Rezultat, ki smo ga dobili s to metodo, je presenetljiv in dokazuje, da slike res vsebujejo mnogo informacij o globini. Te informacije lahko uporabimo, da določimo globino tudi, če nimamo stereo pogleda; ali kot dodatno informacijo pri razločevanju dvoumnosti. Vendar pa takšen pristop ni najbolj zanesljiv in kot je razvidno s primerov na slikah 55 do 57, rezultati ne dajejo povsem točne ali prave informacije o globini.

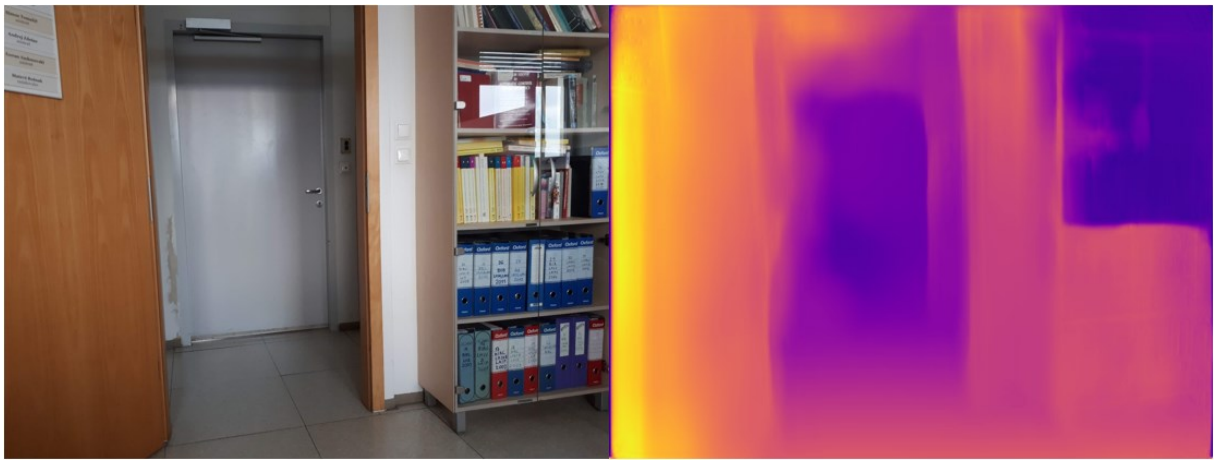


Slika 55. Primer 1: Globinska slika iz posamičnega prizora pridobljena z nevronsko mrežo *monodepth*



Slika 56. Primer 2: Globinska slika iz posamičnega prizora pridobljena z nevronsko mrežo *monodepth*

Na sliki 57 vidimo, da je v zgornjem delu slike prišlo do nepričakovanega izračuna globine, ki je verjetno posledica odseva v steklenih vratih omare. V tem primeru je morda izračunana dispariteta veljavna za odboj oz. objekte za kamero in dobljeni rezultat ni popolnoma napačen.



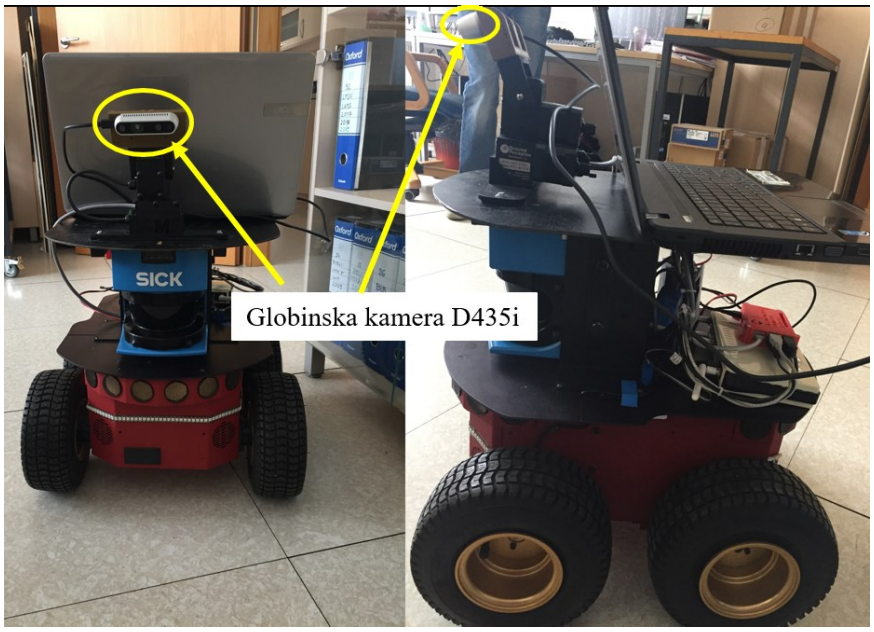
Slika 57. Primer 3: Globinska slika iz posamičnega prizora pridobljena z nevronsko mrežo *monodepth*

3.3 Zaznavanje ovir v oblaku točk

V tem poglavju bomo predstavili problem in rezultate zaznavanja tal z uporabo globinske kamere, ki temelji na stereo vidu. Najprej bomo definirali kaj sploh so ovire in na kakšen način smo vršili zaznavanje tal. Da bi lahko segmentirali okolje na tla in ovire z globinsko kamero, je potrebno izračunati oblak točk iz globinske slike, ki ga bomo opisali v podpoglavju 3.5.1.

3.4 Predstavitev problema

Z uporabo globinske kamere Intel RealSense D435i želimo zaznavati ovire in tla in na ta način robotu omogočiti predstavo o okolju v katerem se nahaja. Kamera je nameščena na robota Pioneer 3-AT, kot je prikazano na sliki 58.



Slika 58. Robot Pioneer 3-AT in globinska kamera RealSense D435i

Celoten sistem, poleg kamere in robota, sestavljajo še brezžični usmerjevalnik, mikroračunalnik Raspberry Pi in prenosni računalnik (slika 59).

Brezžični usmerjevalnik omogoča komunikacijo med računalnikom in robotom oz. vgradnim sistemom Raspberry Pi, na katerem teče operacijski sistem za robote ROS (angl. *Robot Operating System*). Preko brezžične povezave lahko pošiljamo ukaze s prenosnega računalnika na Raspberry Pi in upravljamo robota.

Kamera ja preko USB-kabla tipa C povezana s prenosnim računalnikom (slika 60), na katerem poteka celotna obdelava globinskih podatkov s kamere v programskem orodju Matlab. Podatke iz kamere pretvorimo v oblak točk in segmentiramo okolje na tla in ovire. Ker je v kamero vključena inercialna merilna enota IMU, ki vključuje pospešek in žiroskop, moramo kamero priključiti na USB-priključek, ki podpira verzijo USB 3.1. V nasprotnem primeru ne

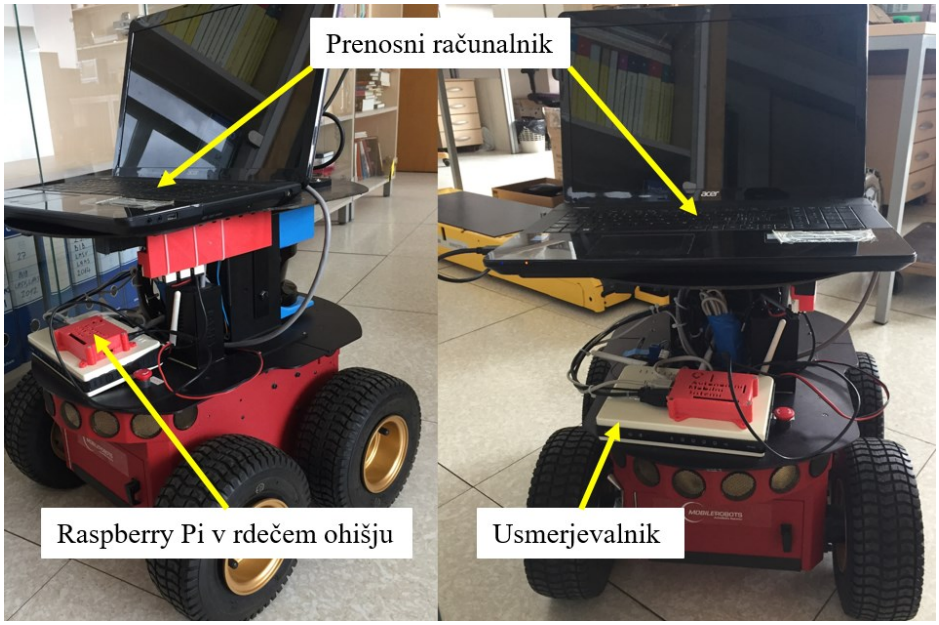
moremo pridobivati podatkov s pospeškometra ali žiroskopa. Število slik na sekundo, ki jih dobivamo iz kamere in resolucija le-teh je prav tako odvisna od povezave kamere na tip USB-priključka (USB 2.0 ali USB 3.1) na računalniku. V tabelah 4 in 5 so podani podatki o številu slik na sekundo in resolucija slik pri povezavi kamere na določen tip USB-priključka.

Tabela 4. Resolucija slike in število slik na sekundo (USB 3.1)

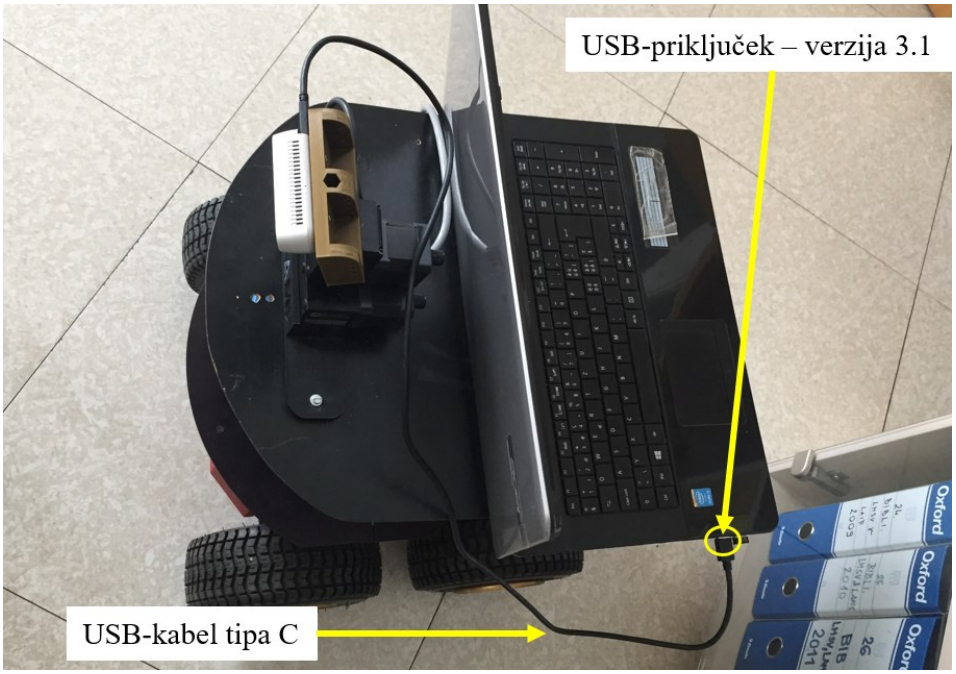
Format	**Resolucija**	**Število slik na sekundo [Hz]**	**Komentar**
Z [16 bitov]	1280 × 720	6, 15, 30	Globinska slika
	848 × 480	6, 15, 30, 60, 90	
	640 × 480	6, 15, 30, 60, 90	
	640 × 360	6, 15, 30, 60, 90	
	480 × 270	6, 15, 30, 60, 90	
	424 × 240	6, 15, 30, 60, 90	
Kalibracija		15, 25	Kamere: D400/D410/D415
		15, 25	Kamere: D420/D430/D435/D435i

Tabela 5. Resolucija slike in število slik na sekundo (USB 2.0)

Format	**Resolucija**	**Število slik na sekundo [Hz]**	**Komentar**
Z [16 bitov]	1280 × 720	6	Globinska slika
	640 × 480	6, 15, 30	
	480 × 270	6, 15, 30, 60	



Slika 59. Ostale komponente mobilnega sistema



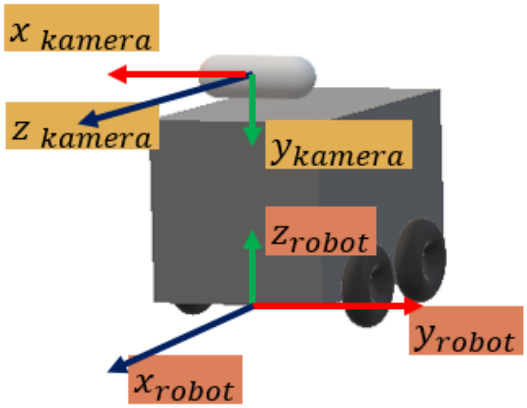
Slika 60. Povezava kamere in prenosnega računalnika preko USB kabla tipa C

Nagib oz. orientacijo kamere na robotu lahko spreminjamo. To je pomembno predvsem pri sistemu za vodenje robota oz. preprečevanju trkov, ker želimo oviro zaznati v pravem trenutku na ustrezni razdalji. Vidno polje stereo kamere FOV (angl. *Field Of View*) je definirano s prekrivanjem vidnega polja leve in desne kamere in je podano v tabeli 6.

Tabela 6. Vidno polje globine, ki je določeno na razdalji dveh metrov.

Format	D435i
Horizontalni FOV (VGA 4:3)	74
Vertikalni FOV (VGA 4:3)	62
Diagonalni FOV (VGA 4:3)	88
Horizontalni FOV (HD 16:9)	86
Vertikalni FOV (HD 16:9)	57
Diagonalni FOV (HD 16:9)	94

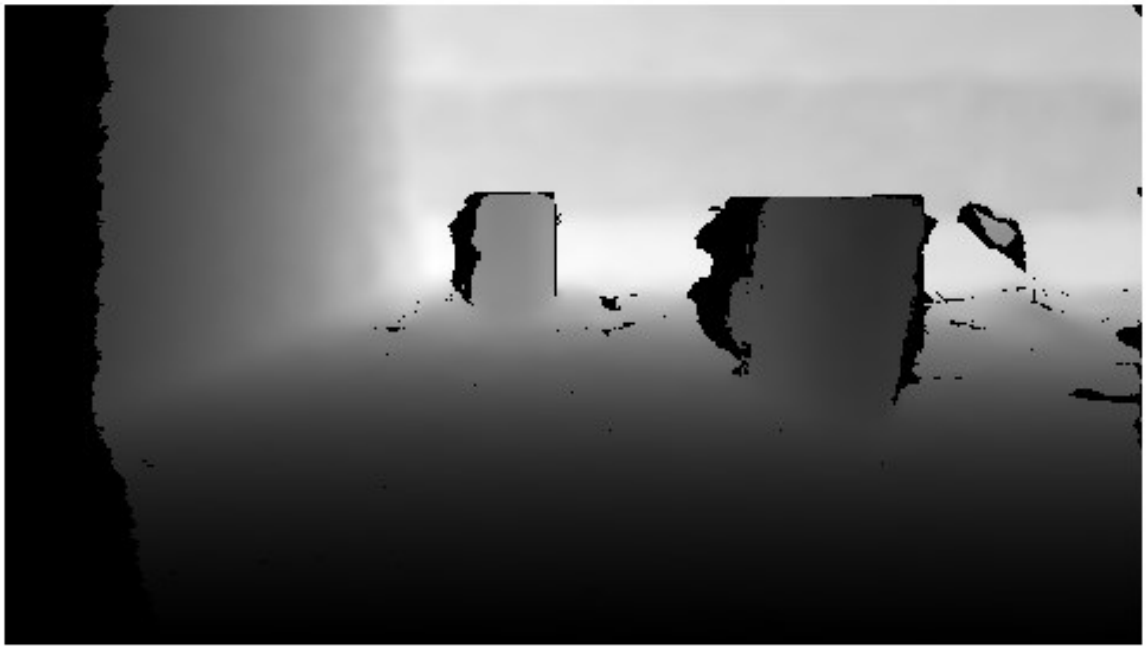
Koordinatni sistem kamere in robota je prikazan na sliki 61. Detekcijo tal smo vršili v koordinatnem sistemu kamere, nato smo tridimenzionalne podatke preslikali v koordinatni sistem robota, kjer smo vršili detekcijo ovir.



Slika 61. Koordinatni sistem robota in kamere

S kamero D435i zajemamo v Matlab globinske slike z resolucijo 480 × 270, kjer vsak slikovni element vsebuje informacijo o globini (dejansko razdaljo). Na sliki 62 je prikazan posnet prizor s katerim bomo opisali obdelavo oblaka točk in segmentacijo okolja na ovire in tla. Na sliki 63 je prikazana sivinska slika, ki vsebuje informacijo o globini. Svetlejši predeli predstavljajo večjo oddaljenost od kamere, temnejši pa manjšo. Črna območja na globinski sliki predstavljajo slikovne elemente, za katere kamera ni uspela izračunati razdalje. Vzrok za to je lahko, da določene točke v prostoru niso zaznane s strani obeh kamer in posledično ne moremo najti para točk na slikah leve in desne kamere. Problem lahko predstavlja tudi algoritem za iskanje parov, ki ne uspe določiti vseh parov, četudi obe kameri zaznata isto točko v prizoru.

Slika 62. Posneti prizor za obdelavo oblaka točk in segmentacijo okolja



Slika 63. Pripadajoča globinska slika posnetega prizora

Razlog za izbiro manjše resolucije slike je v tem, ker se z zmanjševanjem resolucije manjša razdalja, na kateri lahko s kamero še zaznamo objekte.

3.5 Kalibracija sistema za detekcijo ovir

Prvi korak pri kalibraciji sistema je avtomatska detekcija orientacije kamere oz. koordinatnega sistema kamere, v katerem bomo vršili detekcijo tal. Temu sledi pretvorba koordinatnega sistema kamere v koordinatni sistem robota. V prvem koraku kalibracije, kjer vršimo detekcijo tal, je pomembno, da se robot nahaja v praznem prostoru. V tem primeru bo večino tridimenzionalnih točk pripadala tlom. V podpoglavju 3.5.2 je predstavljeno zaznavanje tal s histogramom. Kalibracijo sistema ne izvajamo v vsakem trenutku, temveč na začetku, saj predpostavljamo, da je kamera glede na robota statična. Avtomatsko kalibriranje takšnih sistemov je zelo pomembno, saj na ta način lahko poenostavimo delo z mobilnimi sistemi v praksi. V tem primeru lahko s pritiskom na gumb omogočimo sistemu zavedanje o okolici oz. kje se nahajajo tla glede na poljubno orientacijo kamere. V podpoglavju 3.5.3 je predstavljena segmentacija okolja pred robotom.

3.5.1 Obdelava oblaka točk

Ker iz kamere dobimo sivinsko sliko (z resolucijo 480×270), katere vsak slikovni element vsebuje podatek o globini oz. razdalji do točke v okolju, lahko sestavimo tridimenzionalno sliko okolja s triangulacijo. Postopek izračuna oblaka točk je po korakih opisan v algoritmu 1.

Algoritem 1 Psevdokoda za izračun oblaka točk

1: Inicializacija intrinzičnih parametrov f_x, f_y, c_x, c_y

2: Inicializacija globinske slike G

3: Inicializacija števila vrstic slike M in števila stolpcev slike N

4: **za** $i = 1,2, \dots, M$ ponavljaj

5: **za** $j = 1,2, \dots, N$ ponavljaj

6: Izračun 3D-koordinat: $x = (j - c_x)\, G(i,j) \,/\, f_x$

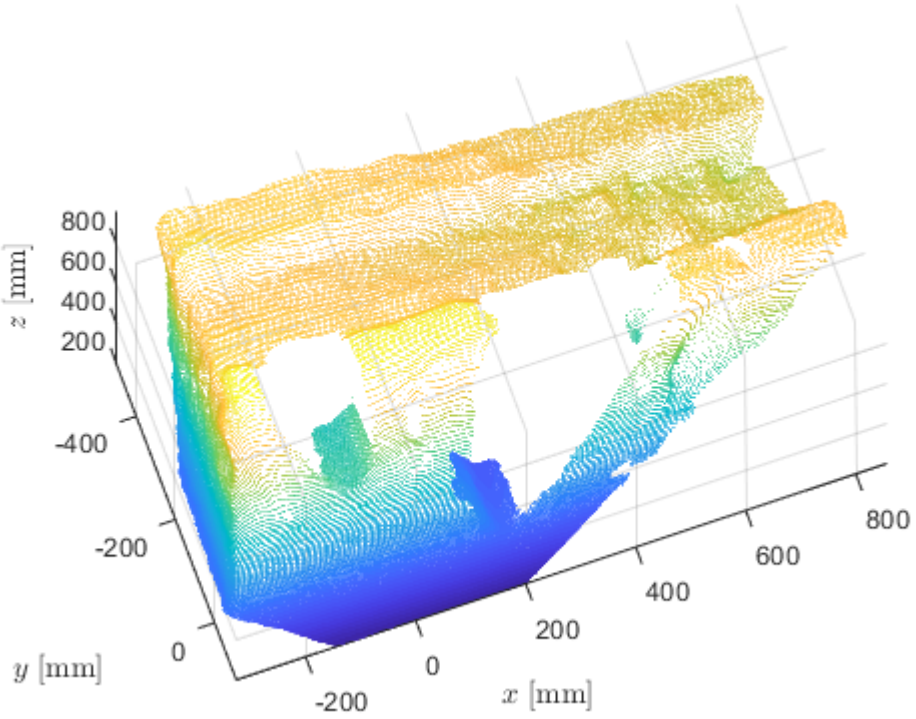
7: $y = (i - c_y)\, G(i,j) \,/\, f_y$

8: $z = G(i,j)$

9: **konec**

10: **konec**

Na sliki 64 je prikazan primer oblaka točk, ki smo ga izračunali s triangulacijo.

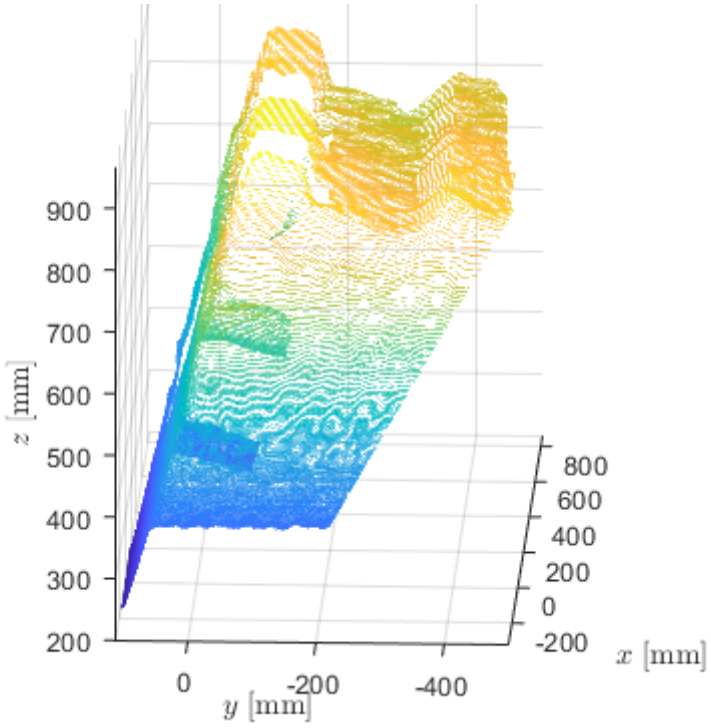


Slika 64. Oblak točk

Ker ima kamera določen nagib glede na njeno postavitev na robotu, je oblak točk rotiran okrog osi za vrednosti Eulerjevih kotov:

- φ oz. nagib (angl. *roll*), ki pomeni rotacijo okrog x-osi,
- θ oz. naklon (angl. *pitch*), ki pomeni rotacijo okrog y-osi in
- ψ oz. odklon (angl. *yaw*), ki pomeni rotacijo okrog z-osi.

Na sliki 65 je prikazan oblak točk, ki je rotiran okrog osi x.



Slika 65. Neporavnan oblak točk

Ker predpostavljamo, da je robot na ravni podlagi, katere normala je vzporedna s silo gravitacije, je potrebno za namen detekcije tal poravnati oblak točk. Tako bo y-os le-tega poravnana s silo gravitacije. Ker je v kamero vključena enota IMU, lahko s podatki iz pospeškometra in žiroskopa ugotovimo, za kolikšen kot je kamera rotirana okoli svojih osi. Ker nas zanima le poravnanost y-osi s silo gravitacije, potrebujemo le vrednosti kotov φ in ψ. Tako lahko sestavimo rotacijski matriki R_x in R_z (dimenzije 3×3), s katerima bomo rotirali oblak točk v želeno orientacijo

$$\boldsymbol{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & \sin(\varphi) \\ 0 & -\sin(\varphi) & \cos(\varphi) \end{bmatrix} \tag{27}$$

$$\boldsymbol{R}_z = \begin{bmatrix} \cos(\varphi) & \sin(\varphi) & 0 \\ -\sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{28}$$

Kota φ in ψ lahko izračunamo iz komponent vektorja pospeška $\underline{\boldsymbol{a}} \in \mathbb{R}^3$, ki ga dobimo iz pospeškometra in je zapisan z enačbo

$$\underline{\boldsymbol{a}}^T = [a_x \; a_y \; a_z]. \tag{29}$$

Vektor pospeška je povprečje več meritev, ki jih pridobimo iz enote IMU v globinski kameri, medtem ko čakamo, da se prenos slike iz kamere umiri. Podobno kot pri detekciji tal, tudi računanje rotacijskih matrik izvedemo enkrat. Kot φ izračunamo po enačbi

$$\varphi = \text{asin}\left(\frac{a_z}{a_y}\right), \tag{30}$$

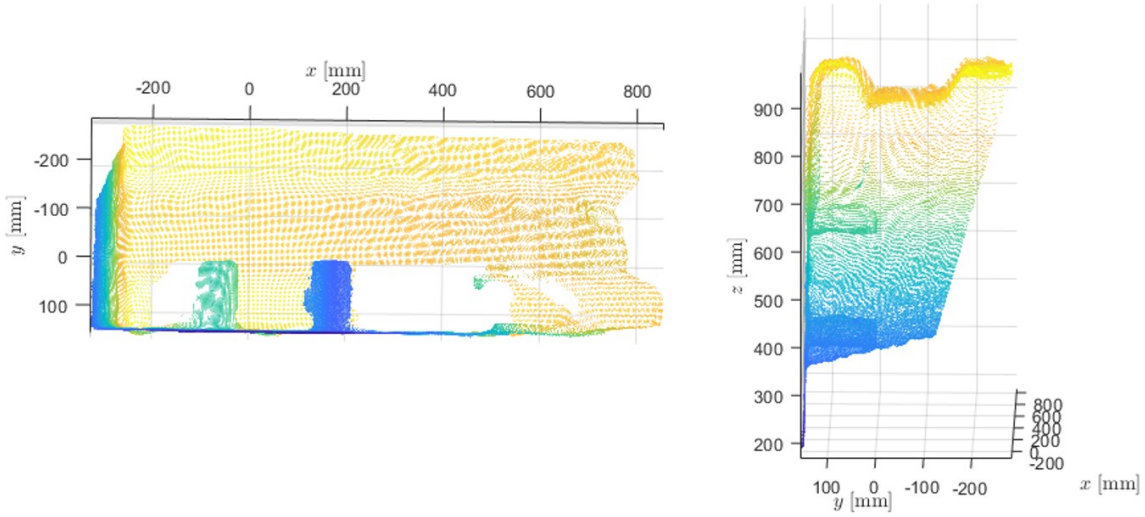
kot ψ pa po enačbi

$$\psi = -atan2\left(\frac{a_x}{a_y}\right) + \pi. \tag{31}$$

Rotacijo oblaka točk za določene kote izvedemo z enačbo

$$\boldsymbol{D}_{rot} = (\boldsymbol{R}_x \boldsymbol{R}_z \boldsymbol{D}^T)^T, \tag{32}$$

kjer je matrika $\boldsymbol{D}$ oblak točk, dimenzije $N \times 3$ (N je število 3D-točk v $\boldsymbol{D}$) in matrika $\boldsymbol{D}_{rot}$ poravnan oblak točk (slika 66), dimenzij $N \times 3$, kjer vsak stolpec pripada koordinatam (x,y,z).

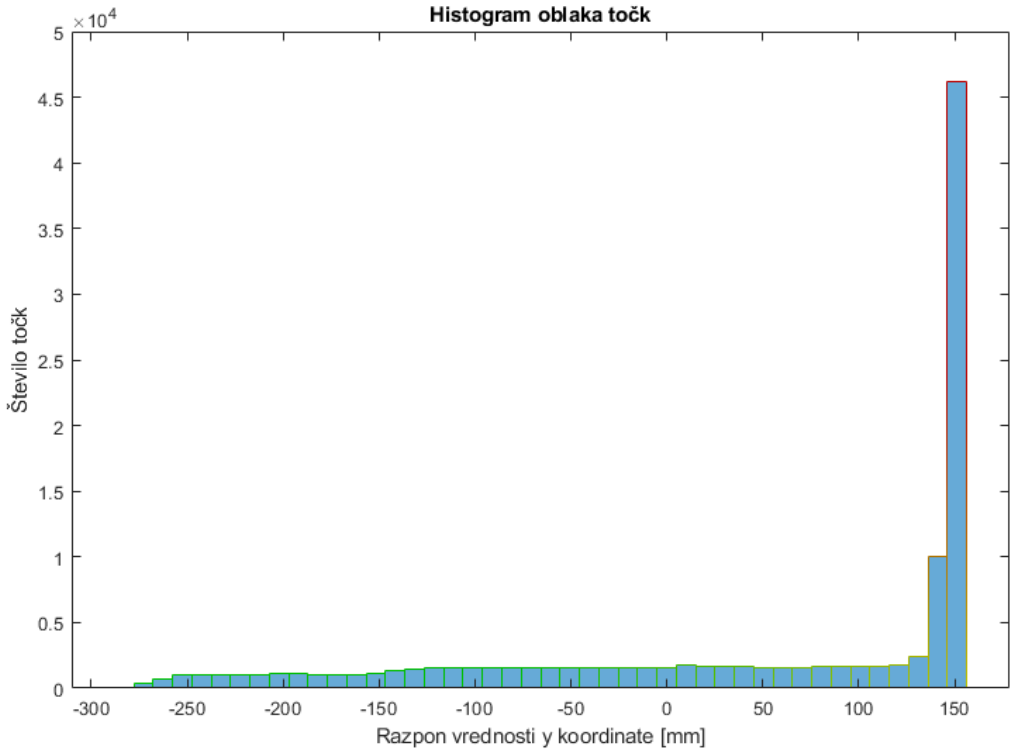


Slika 66. Poravnan oblak točk

3.5.2 Detekcija tal

Tla smo zaznavali v koordinatnem sistemu kamere vzdolž osi y oblaka točk, ki je po postopku kalibracije sistema vzporedna z gravitacijo. Iskanje y-koordinate tal smo opravili s histogramom, ker smo v postopku kalibracije sistema pričakovali, da bo večino točk oblaka pripadalo tlom. Ker se v meritev globine na večjih razdaljah vnaša šum, smo pri izračunu globine upoštevali standardno deviacijo oz. razpršenost točk tal, ki jo z večanjem opazovane, merjene globine povečujemo. Standardno deviacijo smo določili eksperimentalno.

Slika 67 prikazuje histogram oblaka točk prizora, ki smo ga posneli in prikazali na sliki 62. Ker smo kalibracijo sistema izvedli v prostoru, kjer večina točk pripada tlom, se tla nahajajo na razdalji, kjer ima histogram maksimum.

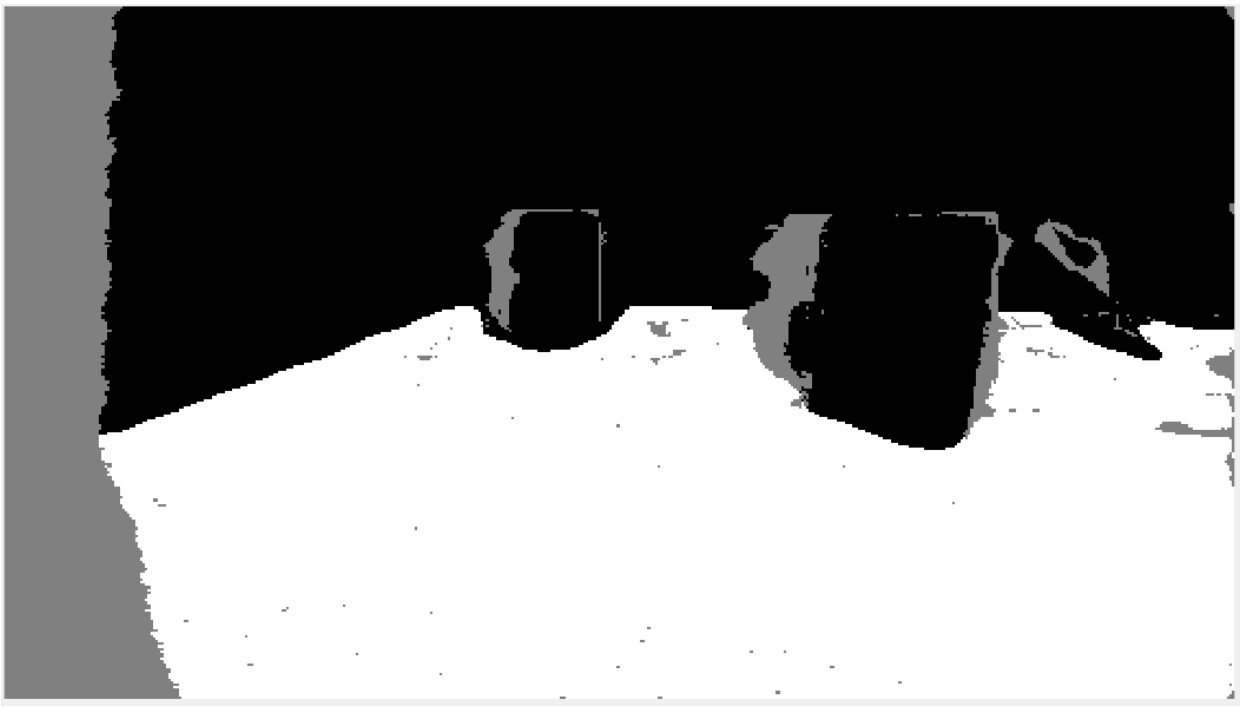


Slika 67. Histogram oblaka točk

Informacija o položaju tal na y-osi je pomembna pri preslikavi oz. translaciji koordinatnega sistema kamere v koordinatni sistem robota.

Ko imamo narejeno detekcijo tal, lahko dan prizor segmentiramo na tla in ovire, pri čemer upoštevamo standardno deviacijo tal, ki je v našem primeru znašala $s = 15\,mm$. Segmentacija v tem primeru enostavno pomeni, da so ovire vse kar niso tla. Na sliki 68 je prikazan rezultat segmentacije posnetega prizora, kjer:

- siva območja predstavljajo točke, kjer nimamo informacije o okolju (neznano),
- bela območja predstavljajo tla (prosto) in
- črna območja predstavljajo ovire.



Slika 68. Detekcija tal

Postopek kalibracije s poravnavo oblaka točk in detekcijo tal omogoča, da lahko avtomatično določimo kje so tla, ne glede na orientacijo in položaj kamere.

Ker je kamera nameščena na robotu in ker je cilj vodenje robota, bomo oblak točk $\boldsymbol{D}_{rot}$ preslikali v koordinatni sistem robota, kjer os x kaže v smeri vožnje, z-os je pravokotna na podlago oz. vzporedna z vektorjem gravitacije, y-os pa je definirana tako, da dobimo desnosučni koordinatni sistem. Preslikavo koordinatnega sistema naredimo z enačbo

$$\boldsymbol{D}_{robot} = \boldsymbol{D}_{rot}\boldsymbol{R} + \boldsymbol{T}, \tag{33}$$

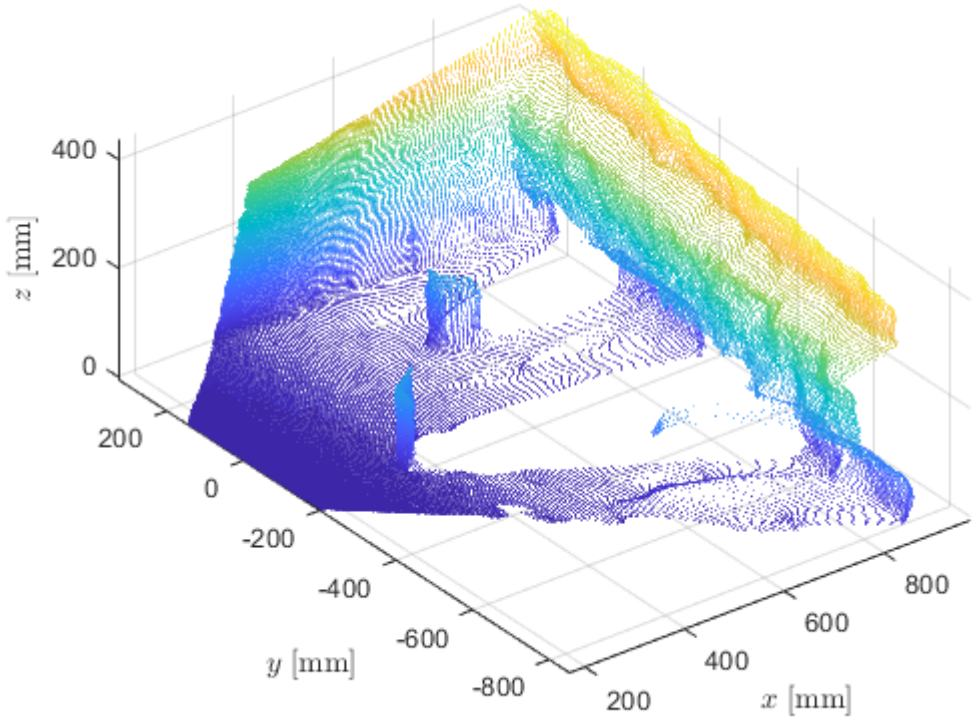
kjer je $\boldsymbol{R}$ rotacijska matrika dimenzije 3×3

$$\boldsymbol{R} = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix} \tag{34}$$

in $\boldsymbol{T}$ translacijski vektor dimenzije $N \times 3$

$$\boldsymbol{T} = \begin{bmatrix} 0 & 0 & t_1 \\ 0 & 0 & t_2 \\ \vdots & \vdots & \vdots \\ 0 & 0 & t_N \end{bmatrix}, \tag{35}$$

kjer so elementi v $\boldsymbol{T}$, ki so različni od nič, enaki vrednostim spremenljivke tla, $t_1 = t_2 = \cdots = t_N = tla$. Tla se tako nahajajo v ravnini $z = 0$ (slika 69).



Slika 69. Oblak točk v koordinatnem sistemu robota

3.5.3 Segmentacija okolja

Ker nas zanima le okolica pred robotom, jo bomo definirali kot $0 < x < x_{max}$ in $-y_{max} < y < y_{max}$. To pomeni, da gradimo dvodimenzionalni, lokalni zemljevid robota. Z gradnjo lokalnega zemljevida se želimo izogniti vnosu šuma v meritve globine na večjih razdaljah. Okolico bomo diskretizirali in na sliki prikazali ovire, prosto in neznano še na tem primeru.

Za krajni vrednosti območja danega prizora smo izbrali $x_{max} = 1\ m$ in $y_{max} = 0.5\ m$, ki smo ga diskretizirali s korakom po pet milimetrov. V sistemih vodenja robota bomo za krajni vrednosti izbrali vrednosti $x_{max} = 2\ m$ in $y_{max} = 1\ m$, ker potrebujemo več informacije o okolici pred robotom. Postopek gradnje lokalnega zemljevida je opisan v algoritmu 2.

Algoritem 2 Psevdokoda za gradnjo lokalnega zemljevida robota

1: Izberi krajni vrednosti območja zemljevida x_{max} in y_{max}

2: Inicializiraj točke lokalnega zemljevida iz oblaka točk $x_{okolica}$, $y_{okolica}$ in $z_{okolica}$

3: Izbira koraka diskretizacije k.

4: Diskretiziraj območje s korakom k in določi dolžini območja x_{len} in y_{len}

5: Klasificiraj točke okolice na tiste, ki pripadajo tlom in oviram, x_{ovira} in y_{ovira} ter x_{tla} in y_{tla}

6: Inicializiraj matriko zemljevida robota s sivinskimi vrednostmi $MAP \in x_{len} x\ y_{len}$

7: Prištej y-koordinatam y_{max}, shrani v novo spremenljivko Y_{ovira} in y_{tla}

8: **za** $i = 1,2, \dots, dolzina(x_{tla})$ ponavljaj

9: Računanje lokacije tal v matriki MAP: $vrstica\ =\ x_{len} - floor\left(\frac{x_{tla}(i)}{k}\right)$

10: $$stolpec\ =\ floor(\frac{Y_{tla}(i)}{k})$$

11: Označevanje tal (bela območja) zemljevida: $MAP(vrstica + 1, stolpec + 1)\ =\ 1$

12: **konec**

13: **za** $j = 1,2, \dots, dolzina(x_{ovira})$ ponavljaj

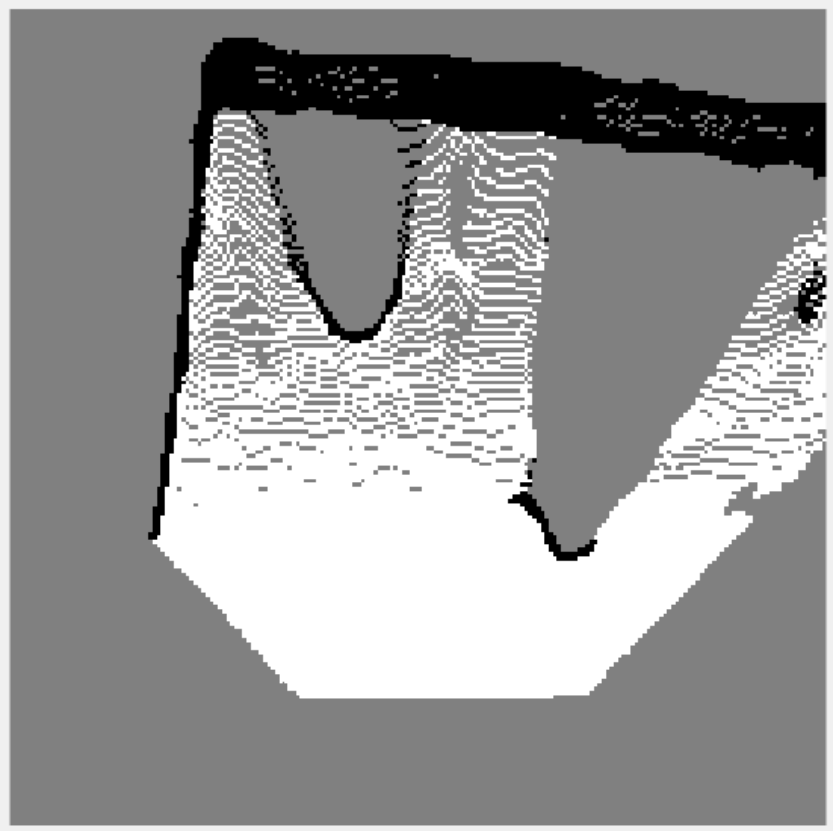
14: Računanje lokacije ovir v matriki MAP: $vrstica\ =\ x_{len} -\ floor\left(\frac{x_{ovira}(j)}{k}\right)$

15: $$stolpec\ =\ floor\left(\frac{Y_{ovira}(j)}{k}\right)$$

16: Označevanje ovir (črna območja) zemljevida: $MAP(vrstica + 1, stolpec + 1)\ =\ 0$

17: **konec**

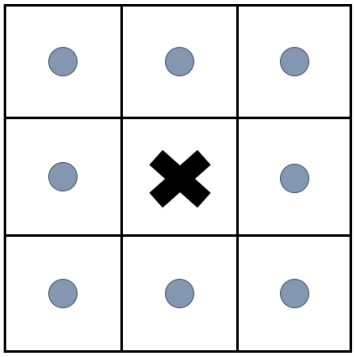
Na sliki 70 je prikazan segmentiran lokalni zemljevid robota (pogled od zgoraj), ki smo ga pridobili z algoritmom za gradnjo zemljevida.



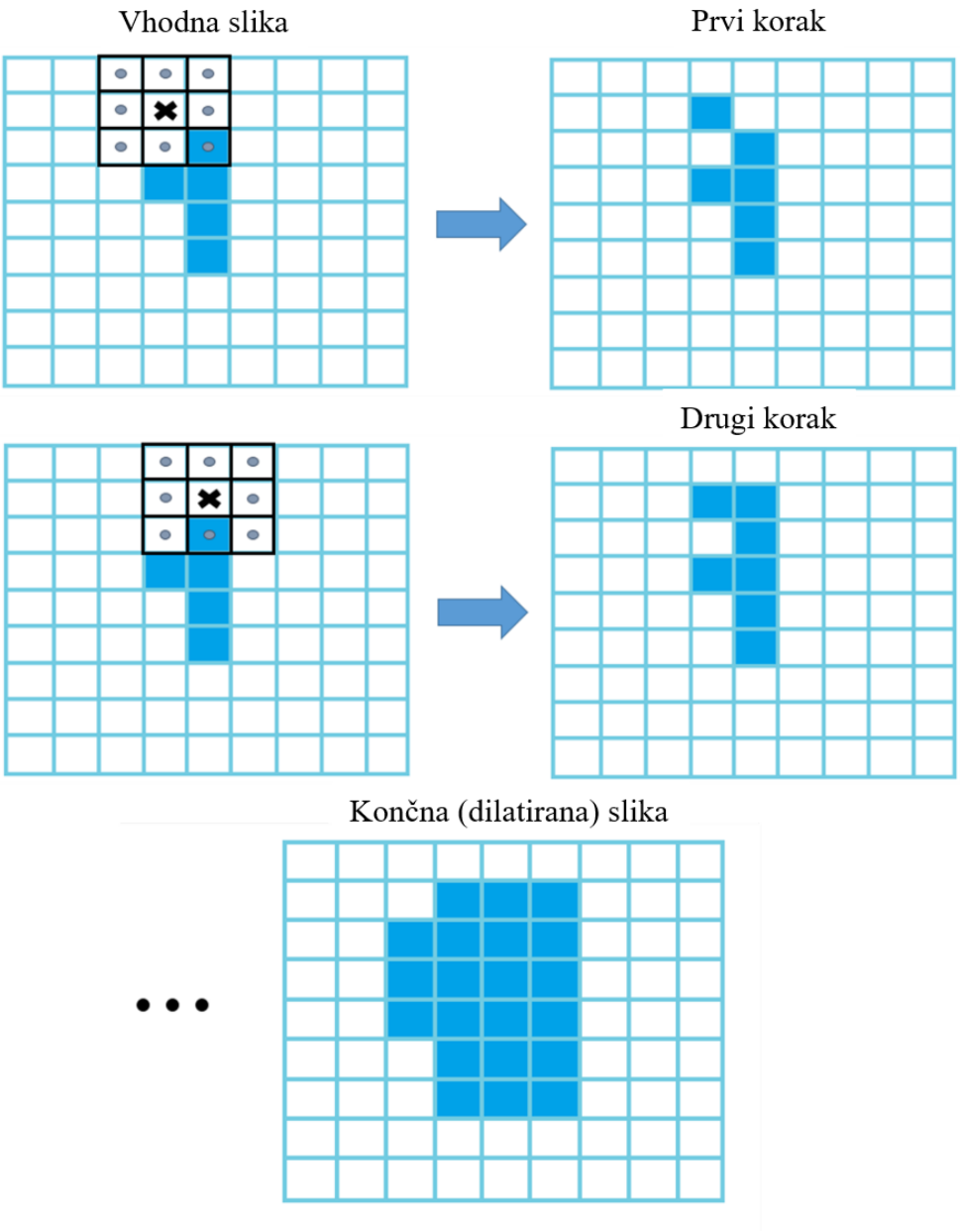
Slika 70. Segmentiran lokalni zemljevid robota

Ker je na zemljevidu robota veliko neznanih predelov, kjer pričakujemo tla, smo najprej zapolnili prostor na področju tal z morfološko operacijo dilatacije in nato s preprostim filtrom lukenj.

Neformalna definicija dilatacije [71] je, če vsaj eden od sosedov v strukturnem elementu pripada objektu (tlom), potem dilatiraj središčno točko elementa (slika 71). To pomeni, da dilatirana točka postane del objekta (slika 72).



Slika 71. Strukturni element morfološke operacije dilatacije



Slika 72. Ilustracija dilatacije

Algoritem dilatacije je po korakih opisan v algoritmu 3.

Algoritem 3 Psevdokoda dilatacije lokalnega zemljevida robota

1: Izbira velikosti strukturnega elementa vrs in st
2: Inicializacija začetnih vrednosti števcev za območja na sliki $siva = 0$, $bela = 0$, $črna = 0$
3: Izbira maksimalne vrednosti števila sivih točk znotraj elementa $siva_{max}$
4: **za** $i = 1,2,\ldots,$ $število\_vrstic(MAP) - vrs$ ponavljaj
5: **za** $j = 1,2,\ldots,$ $število\_stolpcev(MAP) - st$ ponavljaj
6: S strukturnim elementom se premikamo po zemljevidu robota:
$$element = MAP(i: i + vrs - 1, j: j + st - 1)$$
7: **za** $k = 1,2,\ldots,$ $število\_vrstic(element)$
8: **za** $l = 1,2,\ldots,$ $število\_stolpcev(element)$
9: če je $element(k, l)$ enak sivinski vrednosti
10: štejemo število sivih točk znotraj elementa: $siva = siva + 1$
11: če je $element(k, l)$ enak vrednosti bele barve
12: štejemo število belih točk znotraj elementa: $bela = bela + 1$
13: če je $element(k, l)$ enak vrednosti črne barve
14: štejemo število črnih točk znotraj elementa: $črna = črna + 1$
15: **konec**
16: **konec**
17: če je $črna = 0$ in $bela > 0$ in $siva < siva_{max}$
18: Postavi središčno točko elementa na vrednost bele barve:
$$MAP(i: i + vris - 2, j: j + st - 2) = 1$$
19: Postavi vrednosti števcev na nič: $siva = 0$, $bela = 0$, $črna = 0$
20: **konec**
21: **konec**

Na sliki 73 je prikazan rezultat operacije dilatacije, kjer smo za vrednosti vhodnih spremenljivk v algoritem dilatacije izbrali:

- $vrs = 3$
- $st = 3$
- $siva_{max} = 9$

Slika 73. Rezultat dilatacije

Algoritem filter lukenj je po korakih opisan v algoritmu 4.

Algoritem 4 Psevdokoda algoritma filter lukenj lokalnega zemljevida robota

1: Izberemo velikosti strukturnega elementa a in b

2: Inicializacija št. točk znotraj elementa $num = ab$

3: Izbira maksimalne vrednosti števila sivih točk znotraj elementa $siva_{max}$

4: Inicializacija začetnih vrednosti števcev za območja na sliki $siva = 0$, $bela = 0$, $črna = 0$

5: **za** $i = 1{:}\, a{:}\, število\_vrstic(MAP) - a$ ponavljaj

6: **za** $j = 1{:}\, b{:}\, število\_stolpcev(MAP) - b$ ponavljaj

7: S strukturnim elementom se premikamo po zemljevidu robota:

$$element = MAP(i + 1{:}\, i + a, j + 1{:}\, j + b)$$

8: **za** $k = 1,2,\ldots, število\_vrstic(element)$ ponavljaj

9: **za** $l = 1,2,\ldots, število\_stolpcev(element)$ ponavljaj

10: če je $element(k, l)$ enak sivinski vrednosti

11: štejemo število sivih točk znotraj elementa: $siva = siva + 1$

12: če je $element(k, l)$ enak vrednosti bele barve

13: štejemo število belih točk znotraj elementa: $bela = bela + 1$

14: če je $element(k, l)$ enak vrednosti črne barve

15: štejemo število črnih točk znotraj elementa: $črna = črna + 1$

16: **konec**

17: **konec**

18: če je $siva = num$

19: Preskoči pogojni stavek in nadaljuj z izvajanjem zanke

20: če je $bela > 0$ in $črna = 0$ in $siva < siva_{max}$

21: Postavi vse točke v elementu na vrednost bele točke:

$$MAP(i + 1{:}\, i + a, j + 1{:}\, j + b) = 1$$

22: Postavi vrednosti števcev na nič: $siva = 0$, $bela = 0$, $črna = 0$

23: **konec**

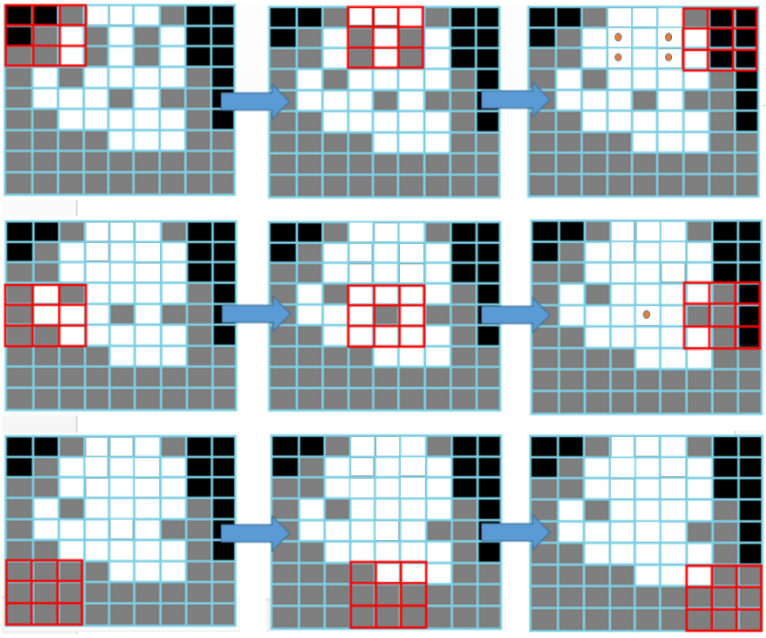
24: **konec**

Na sliki 74 je prikazan rezultat algoritma filter lukenj, ki smo ga uporabili po dilataciji slike zemljevida robota, kjer smo za vhodne vrednosti spremenljivk v algoritem filter lukenj izbrali:

- $a = 3$
- $b = 3$
- $siva_{max} = 5$

Slika 74. Rezultat filtra lukenj

Na sliki 75 je predstavljen postopek delovanja algoritma filtra lukenj na enostavnem primeru, kjer smo upoštevali enake vhodne vrednosti spremenljivk.



Slika 75. Postopek delovanja filtra lukenj

Na dobljenem zemljevidu robota je veliko črnih točk oz. točk, ki predstavljajo ovire (slika 74), od katerih za zaznavanje trkov potrebujemo le tiste, ki so najbližje robotu. V algoritmu 5 je po korakih opisana psevdokoda algoritma prosti žarek, ki odstrani tiste točke, ki so odvečne pri računanju morebitnih trkov v sistemih za preprečevanje trkov. Ideja algoritma je, da v vsako vidno smer usmerimo žarek in poiščemo prvo točko, kjer se žarek odbije od ovire ali neznanega območja. Pri tem omenimo, da črne točke, ki se nahajajo tik pred robotom (na razdalji manjši od x_{min}) v sivem predelu zemljevida, ne odstranimo pri izvajanju algoritma prosti žarek.

Algoritem 5 Psevdokoda algoritma prosti žarek

1: Nastavimo minimalno razdaljo x_{min}

2: Nastavimo minimalni in maksimalni vidni kot α_{min} in α_{max}

1: Transformiramo vhodni zemljevid robota v polarne koordinate, kjer je na abscisni osi vidni kot α in na ordinatni osi pa razdalja d

2: **za** vidne kote $\alpha_{min} \leq \alpha \leq \alpha_{max}$,

3: glede na x_{min} poiščemo izhodišče žarka in označimo izhodiščno razdaljo z d_0

4: nastavimo vrednost spremenljivke d_1 na maksimalno razdaljo d_{max}

5: v območju med razdaljama d_0 in d_{max} poiščemo najmanjšo razdaljo, ki ne pripada tlom, in priredimo to vrednost razdalje spremenljivki d_1

6: vse točke v območju $d_0 \leq d < d_1$ označimo kot proste (bela barva)

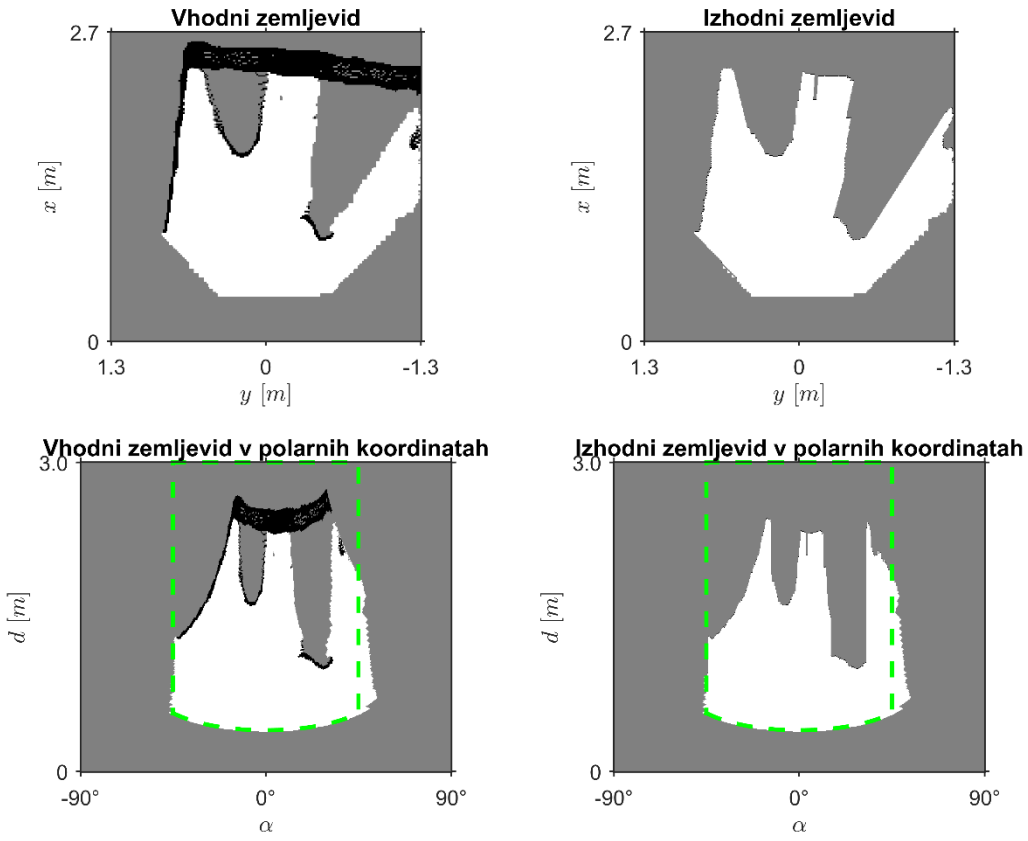
7: točko d_1 označimo, da je ovira (črna barva)

8: vse točke v območju $d_1 \leq d < d_{max}$ označimo kot neznano območje (siva barva)

9: **konec**

10: Transformiramo zemljevid nazaj iz polarnih koordinat v kartezične koordinate

Na sliki 76 je prikazan postopek delovanja algoritma prosti žarek



Slika 76. Postopek delovanja algoritma prosti žarek

Z algoritmom prosti žarek dosežemo to, da upoštevamo tudi neznane predele lokalnega zemljevida robota v sistemih za vodenje robota. Ta podatek nam pride prav v primerih, kjer se pred robotom nahajajo stopnice ali druge prepreke, ki jih globinska kamera oz. robot ni zaznal. Na sliki 77 je pobliže prikazan rezultat algoritma prosti žarek, kjer smo za vhod v algoritem izbrali zemljevid, ki je prikazan na sliki 74. Opazimo znatno zmanjšanje števila črnih točk, ki predstavljajo ovire.

Slika 77. Povečana slika zemljevida robota (prosti žarek)

4 Izogibanje oviram

V tem poglavju so predstavljene metode vodenja mobilnega sistema za varno navigacijo na podlagi lokalnega zemljevida, ki je predstavljen v tretjem poglavju. Osredotočili smo se na metode preprečevanja trkov in metodo za iskanje obvozov. V poglavju 4.1 je predstavljen problem izogibanja oviram in preprečevanja trkov. V poglavju 4.2 je opisan model mobilnega robota Pioneer 3-AT in predstavljen sistem za preprečevanje trkov pri vožnji naravnost ter pri zavijanju. Rezultati delovanja implementiranih algoritmov so predstavljeni v poglavju 4.3.

4.1 Predstavitev problema

Avtonomni mobilni sistemi lahko pri izpolnjevanju različnih nalog naletijo na vrsto neprehodnih ovir, ki se jim morajo izogniti ali celo preprečiti morebitni trk. Na podlagi lokalnega zemljevida, kjer zasedena območja predstavljajo ovire, smo izvedli sistem za preprečevanje trkov pri vožnji naravnost in pri zavijanju. Sistem za preprečevanje trkov lahko uporabljamo kot pomožni podsistem pri vodenju robota.

Cilj tovrstnega podsistema je zmanjšanje hitrosti robota ali ustavitev robota in preprečiti morebitni trk z oviro na predhodno določeni razdalji oz. glede na čas do trka. Nadaljnje gibanje robota je mogoče le z odstranitvijo ovire ali z iskanjem obvoza oz. alternativne poti.

Pri vodenju robota bomo obravnavali pravokotno obliko robota, kjer nas bodo zanimali morebitni trki z obrisom robota. Obliko robota lahko tako predstavimo s štirimi daljicami, ki jih uporabimo pri računanju razdalj do ovir ali časov do trka.

4.2 Modeliranje robota

Modeli gibanja mobilnih sistemov omogočajo načrtovanje strategij lokomocije oz. procesa gibanja sistema iz enega mesta na drugo. V tem poglavju je uporabljen kinematični model robota, ki opisuje geometrijske relacije med vhodnimi signali sistema in obnašanjem sistema [2]. To pomeni, da pri opisu gibanja sistema s kinematičnim modelom ne upoštevamo sil in navorov.

Obstaja več različnih kinematičnih modelov, kot so notranja kinematika, zunanja kinematika, direktna kinematika ali inverzna kinematika in omejitve gibanja. Za opis gibanja

je v nadaljevanju predstavljena notranja in zunanja kinematika uporabljenega robota. Notranja kinematika opisuje relacije med notranjimi oz. internimi spremenljivkami robota, npr. vpliv vrtenja koles na gibanje robota. Zunanja kinematika pa opisuje orientacijo in pozicijo vozila glede na referenčni koordinatni sistem.

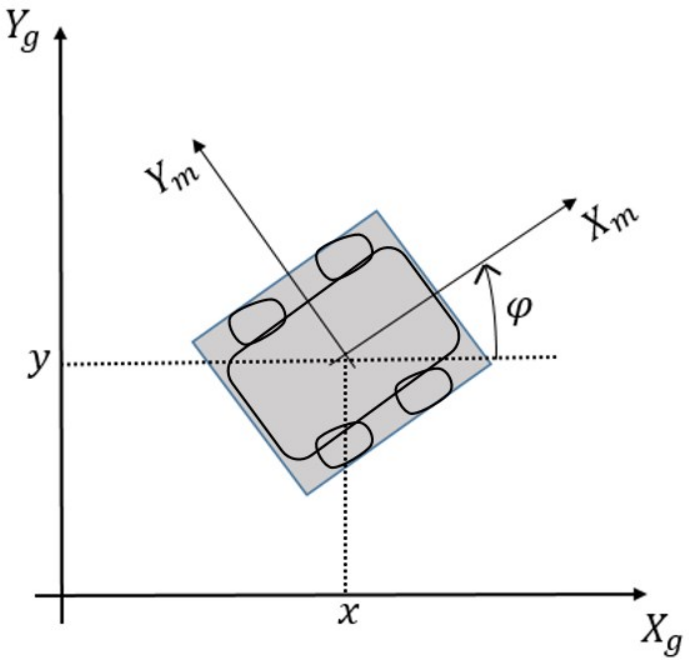
V nadaljevanju je lega robota v ravnini podana z vektorjem stanj

$$\boldsymbol{q}(t) = \begin{bmatrix} x(t) \\ y(t) \\ \varphi(t) \end{bmatrix} \tag{36}$$

v globalnih koordinatah X_g in Y_g kot je prikazano na sliki 78. Koordinatni sistem X_m, Y_m, je pripet na robota in predstavlja lokalni koordinatni sistem. Pri opisu sistema z zunanjo kinematiko lahko podamo relacijo (38) med omenjenima koordinatnima sistemoma z vektorjem translacije $[x, y]^T$ in rotacijsko matriko

$$\boldsymbol{R}(\varphi) = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}, \tag{37}$$

$$\begin{bmatrix} X_g \\ Y_g \end{bmatrix} = \boldsymbol{R}(\varphi) \begin{bmatrix} X_m \\ Y_m \end{bmatrix} + \begin{bmatrix} x \\ y \end{bmatrix} \tag{38}$$



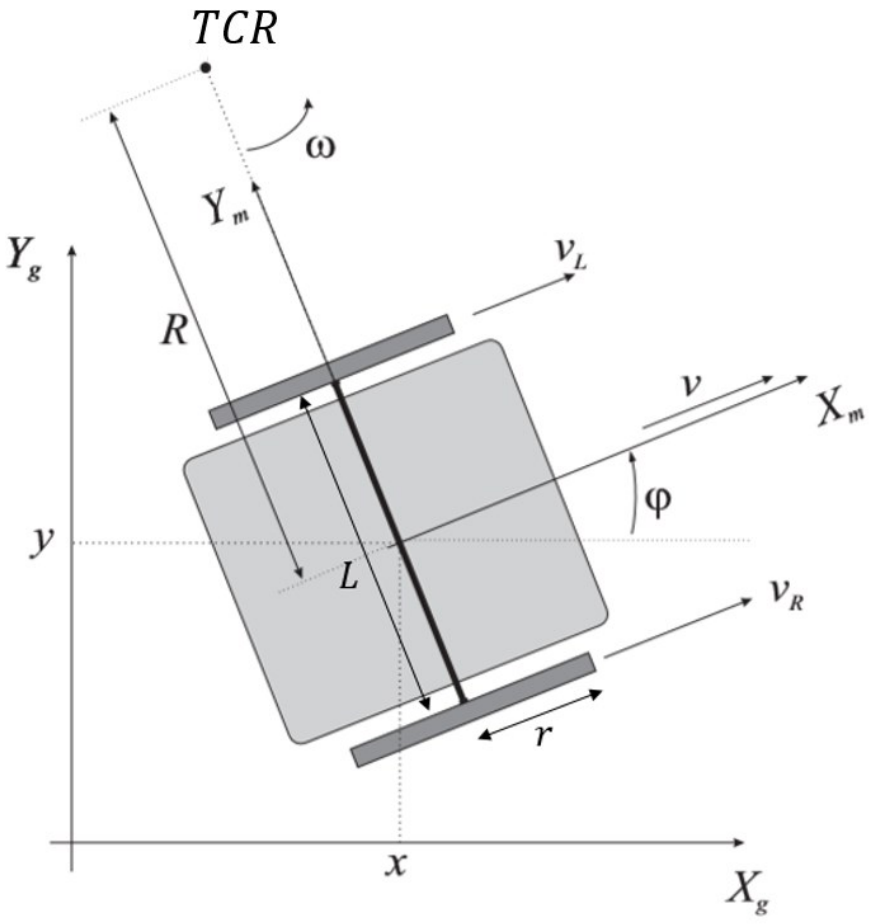
Slika 78. Robot v ravnini

Običajno imajo mobilni roboti kolesa, kjer se lahko vsako kolo prosto vrti okoli svoje osi. Pri opisu gibanja bomo predpostavili model idealnega kotaljenja koles, kjer se kolesa premikajo zaradi kotaljenja in ni zdrsov v smeri kotaljenja ter pravokotno na kotaljenje. Idealno kotaljenje lahko predpostavimo predvsem zaradi zmernih hitrosti, ki jih bo dosegal mobilni

robot. Če kolesa mobilnega robota ne zdrsavajo, se osi vseh koles sekajo v presečišču, ki se imenuje trenutni center rotacije TCR. Okoli presečišča oz. točke TCR krožijo vsa kolesa z enako krožno hitrostjo ω.

Robot Pioneer 3-AT ima štiri kolesa, dva na vsaki strani. Kolesi sta na vsaki strani povezani preko jermena, ki ga poganja motor. Glede na konstrukcijo in pogon koles lahko predpostavimo, da se Pioneer obnaša kot robot z diferencialnim pogonom, zaradi podobnega načina zavijanja. Pri robotu Pioneer je med obračanjem sicer vedno prisotno zdrsavanje koles, a če je zdrsavanje enakomerno porazdeljeno med vsa kolesa, model z diferencialnim pogonom zadostuje.

Na sliki 79 je prikazan primer robota z diferencialnim pogonom, kjer sta vhodni spremenljivki obodni hitrosti levega kolesa $v_L(t)$ in desnega kolesa $v_R(t)$. Pomen parametrov na sliki 79: r je radij kolesa, L je razdalja med kolesoma, $R(t)$ je trenutni radij trajektorije robota in v je translacijska hitrost robota.



Slika 79. Kinematika diferencialnega pogona

Kolesi imata enako krožno hitrost v vsakem trenutku, kar lahko zapišemo z enačbama:

$$\omega = \frac{v_L(t)}{R(t) - \frac{L}{2}}, \tag{39}$$

$$\omega = \frac{v_R(t)}{R(t) + \frac{L}{2}}. \tag{40}$$

Iz zgornjih dveh enačb določimo $R(t)$ in $\omega(t)$:

$$R(t) = \frac{L}{2}\frac{v_R(t) + v_L(t)}{v_R(t) - v_L(t)}, \tag{41}$$

$$\omega(t) = \frac{v_R(t) - v_L(t)}{L}. \tag{42}$$

Translacijsko oz. tangencialno hitrost vozila izračunamo z enačbo:

$$v(t) = \omega(t)R(t) = \frac{v_R(t) + v_L(t)}{2} = \frac{r\omega_R(t) + r\omega_L(t)}{2}, \tag{43}$$

kjer sta $\omega_L(t)$ in $\omega_R(t)$ krožni hitrosti obeh koles okoli njunih osi. Če upoštevamo navedene relacije, lahko zapišemo notranjo kinematiko kot:

$$\begin{bmatrix} x_m(t) \\ y_m(t) \\ \varphi_m(t) \end{bmatrix} = \begin{bmatrix} v_X(t) \\ v_Y(t) \\ \omega(t) \end{bmatrix} = \begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ 0 & 0 \\ \frac{-r}{L} & \frac{r}{L} \end{bmatrix} \begin{bmatrix} \omega_L(t) \\ \omega_R(t) \end{bmatrix}. \tag{44}$$

Zunanja kinematika v globalnih koordinatah je podana z:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\varphi}(t) \end{bmatrix} = \begin{bmatrix} \cos\varphi(t) & 0 \\ \sin\varphi(t) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix}, \tag{45}$$

kjer sta $v(t)$ in $\omega(t)$ vhodni oz. regulirni spremenljivki. Model (45) lahko zapišemo tudi v diskretni obliki, za diskretne čase vzorčenja $t = kT_s$, kjer je T_s čas vzorčenja in $k = 0,1,2, \ldots$:

$$x(k+1) = x(k) + v(k)T_s\cos\varphi(k) \tag{46}$$

$$y(k+1) = y(k) + v(k)T_s\cos\varphi(k) \tag{47}$$

$$\varphi(k+1) = \varphi(k) + \omega(k)T_s \tag{48}$$

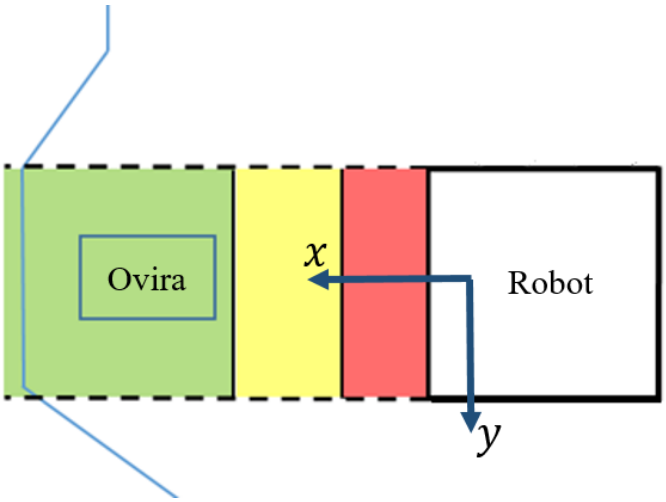
4.2.1 Sistem za preprečevanje trkov

Sistem za preprečevanje trkov deluje kot pomožni podsistem pri navigaciji robota v okolju in preprečuje trke pri vožnji naravnost ter pri zavijanju.

Točke oz. slikovni elementi na lokalnem zemljevidu, ki so označeni s črno barvo, predstavljajo oviro. Za vsak slikovni element imamo zapisano informacijo o lokaciji točke oz. x in y koordinati ovire, ki smo jo pridobili v postopku triangulacije. Ker imamo podano informacijo o lokaciji ovir, lahko v primeru nevarnosti trka ustrezno reagiramo. Koordinate ovir so podane v matriki $\boldsymbol{T}_{ovira}$**:**

$$\boldsymbol{T}_{ovira} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_m & y_m \end{bmatrix} \tag{49}$$

Na sliki 80 je prikazano obnašanje robota pri vožnji naravnost. V postopku preprečevanja trkov so v tem primeru definirana tri območja pred robotom, ki so prikazana na sliki 80. Rdeče območje tik pred robotom je t. i. prepovedano območje. Če se v tem območju nahaja ovira, se mora robot obvezno ustaviti. V rumenem področju robot prilagaja hitrost proporcionalno glede na razdaljo od sprednjega dela robota do ovire. Če se ovira nahaja v zelenem območju, robotu ni potrebno zmanjševati hitrosti in lahko neovirano nadaljuje pot.



Slika 80. Območja detekcije robota pri vožnji naravnost

Algoritem 6 po korakih opisuje delovanje sistema za preprečevanje trkov pri vožnji naravnost.

Algoritem 6 Psevdokoda algoritma za preprečevanje trkov pri vožnji naravnost
1: Izberemo vrednost dolžine d_p, pri kateri začnemo prilagajati hitrost. 2: Izberemo vrednost dolžine d_{stop}, pri kateri ustavimo robota. 3: Poiščemo vse točke $\boldsymbol{T}_{ovira} = [\underline{\boldsymbol{x}}\ \underline{\boldsymbol{y}}]^T \in \mathbb{R}^2 \times \mathbb{R}^N$, ki se nahajajo pred robotom in so znotraj predvidene poti (N je število najdenih točk). Vektorja $\underline{\boldsymbol{x}}$ in $\underline{\boldsymbol{y}}$ z dimenzijo $\mathbb{R}^N$ predstavljata koordinate točk.

4: Poiščemo najmanjšo razdaljo do točk oz. ovir: $d_{min} = \min(\underline{x})$.

5: Če je $d_{min} < d_{stop}$,

6: nastavimo linearno hitrost robota na nič: $v_{zm} = 0$.

7: Če je $d_{stop} \leq d_{min}$ in $d_{min} \leq d_p$,

8: nastavimo linearno hitrost robota glede na d_{min}:

$$v_{zm} = \left(\frac{d_{min} - d_{stop}}{d_p - d_{stop}}\right) v$$

9: Če je $d_{min} > d_p$,

10: ne spreminjamo hitrosti robota: $v_{zm} = v$.

Pri implementaciji sistema za preprečevanje trkov pri zavijanju smo si pomagali z metodo, ki je opisana v članku [72]. Metoda se izvaja v dveh korakih in temelji na računanju časa do trka.

Najprej opravimo detekcijo trka na točkah lokalnega zemljevida robota. Da bi lahko odkrili možnost trka in trenutek, kdaj se bo to zgodilo, sistem predvideva trajektorijo točk v prihodnosti glede na linearno in krožno hitrost robota. Ker smo definirali pravokotno obliko robota, lahko računamo mesta trkov na robotu kot presečišča predvidene trajektorije ovire oz. točke z nizom ravnih črt oz. daljic $D = \{d_1, d_2, \dots, d_n\}$. Ta algoritem je primeren za poljubno obliko robota, saj lahko v tem primeru oris robota predstavimo s poljubnim številom geometrijskih elementov. V nadaljevanju bomo opisali primer računanja časa do trka s sprednjim delom robota, ki ga bomo predstavili z daljico $d_1 = \overline{AB}$. Ostale daljice, ki opisujejo obliko robota so: $d_2 = \overline{AC}$, $d_3 = \overline{CD}$ in $d_4 = \overline{BD}$.

Predvidena trajektorija je krožne oblike (slika 81) in jo izračunamo za vse točke, ki pripadajo oviram $P_o(x_o, y_o)$. Krožna trajektorija je definirana s središčem v $(0, R)$ in radijem $r_o = \sqrt{x_o^2 + (y_o - R)^2}$. Glede na postopek opisan v [73] lahko izračunamo presečišče med daljico $\overline{AB}$, ki je podana s točkama $A(x_a, y_a)$ in $B(x_b, y_b)$, in krogom s središčem v $(0, R)$ ter radijem r_o z naslednjimi enačbami:

$$d_x = x_b - x_a, \qquad d_y = y_b - y_a \tag{50}$$

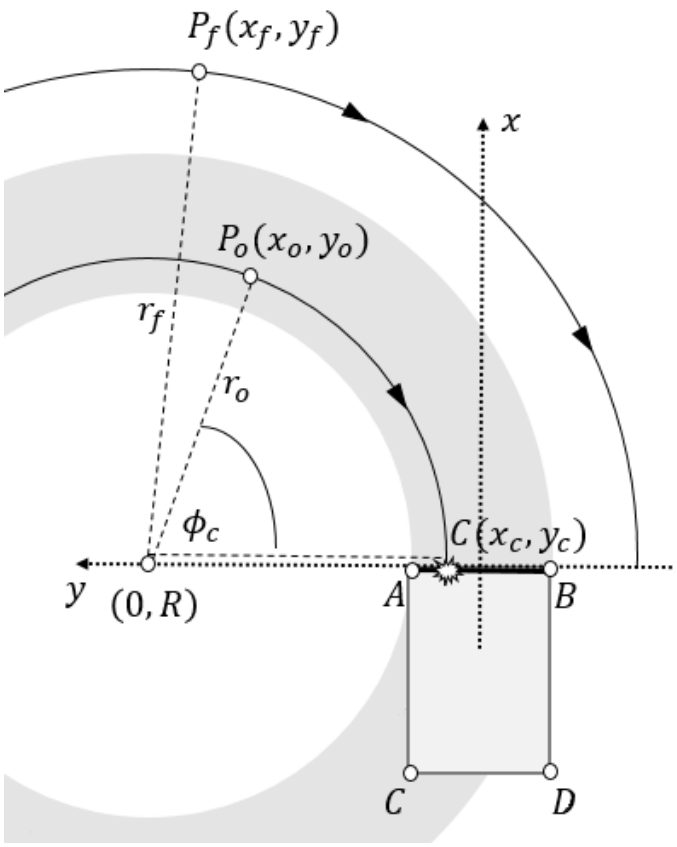
$$d_r = \sqrt{d_x^2 + d_y^2} \tag{51}$$

$$D = \begin{vmatrix} x_a & x_b \\ y_a - R & y_b - R \end{vmatrix} \tag{52}$$

Vrednost diskriminante, ki jo izračunamo z enačbo

$$\Delta = r_o^2 d_r^2 - D^2, \tag{53}$$

določa ali bo prišlo do trka ali ne. Če je $\Delta < 0$, potem presečišča med daljico in krogom ni. Vrednost diskriminante $\Delta = 0$ pomeni, da je daljica tangentna na krog. Pri vrednosti diskriminante $\Delta > 0$ je daljica sekanta, kar pomeni, da obstajata dve presečišči med krogom in daljico. Torej, če obstaja vrednost diskriminante, ki je večja od nič, potem obstaja možnost trka med točko $P_o(x_o, y_o)$ in robotom.



Slika 81. Predvidena trajektorija potovanja in točka trka

Če je $\Delta \geq 0$, lahko točke presečišča oz. trka na robotu izračunamo kot sledi

$$x_c = \frac{D d_y \pm sgn^*(d_y) d_x \sqrt{\Delta}}{d_r^2} \tag{54}$$

$$y_c = R + \frac{-D d_x \pm |d_y| \sqrt{\Delta}}{d_r^2} \tag{55}$$

kjer je $sgn^*(x)$ definiran kot

$$sgn^*(x) = \begin{cases} -1, & x < 0 \\ 1, & sicer \end{cases} \tag{56}$$

Z opisanim pristopom izračunamo tudi presečišča s točkami ovir P_f, ki niso na predvideni trajektoriji trka. Vse točke, ki so na mestu trka z robotom znotraj intervala med točkama A in B, razvrstimo za točke trka. To storimo tako, da vzamemo vse točke z radijem r_o, ki se nahajajo na intervalu med radijem r_a in radijem r_b:

$$r_a = \sqrt{x_a^2 + (y_a - R)^2} \tag{57}$$

$$r_b = \sqrt{x_b^2 + (y_b - R)^2} \tag{58}$$

Ko imamo določene morebitne točke trka C na sprednjem delu robota, lahko izračunamo kot ϕ_c za vsako točko z enačbo

$$\phi_c = \arctan\frac{x_o}{R - y_o} - arctan\frac{x_c}{R - y_c}. \tag{59}$$

Z enačbo (60) lahko ocenimo čase do trka

$$t_{TTC} = \frac{\phi_c}{\omega}, \qquad t_{TTC} \geq 0 \tag{60}$$

Postopek ponovimo za vse daljice, ki sestavljajo obris mobilnega sistema. Vsaki točki na oviri priredimo minimalni čas do trka, med vsemi časi do trka do vsej daljic.

V nadaljevanju je opisan sistem vodenja robota pri zavijanju, ki je bil predstavljen v članku [72]. Algoritem za detekcijo ovir oz. trkov je opisan v algoritmu 7.

Algoritem 7 Psevdokoda algoritma za detekcijo ovir pri zavijanju

1: Inicializacija točke A in točke B.

2: Inicializacija radija r_a in r_b:

$$r_a = \sqrt{x_a^2 + (y_a - R)^2}, \quad r_b = \sqrt{x_b^2 + (y_b - R)^2}.$$

3: Inicializacija linearne hitrosti v in krožne hitrosti ω.

4: Inicializacija središča trajektorije robota $(0, R)$, $R = \frac{v}{\omega}$.

5: Izračunamo radij r_o za točke, ki pripadajo oviram: $r_o = \sqrt{x_o^2 + (y_o - R)^2}$.

6: Izračunamo dolžino d_r daljice $\overline{AB}$ in determinante D:

$$d_x = x_b - x_a, \qquad d_y = y_b - y_a, \qquad d_r = \sqrt{d_x^2 + d_y^2},$$

$$D = \begin{vmatrix} x_a & x_b \\ y_a - R & y_b - R \end{vmatrix}.$$

7: Izračunamo diskriminanto Δ za vse točke, ki pripadajo oviram: $\Delta = r_o^2 d_r^2 - D^2$.

8: Izberemo kandidate točk $C(x_c, y_c)$, za katere velja $\Delta \geq 0$ in $r_A \leq r_o \leq r_B$:

$$x_c = \frac{Dd_y \pm sgn^*(d_y)d_x\sqrt{\Delta}}{d_r^2}, \quad y_c = R + \frac{-Dd_x \pm |d_y|\sqrt{\Delta}}{d_r^2}.$$

9: Za vsako točko trka C izračunamo kot ϕ_c: $\phi_c = \arctan\frac{x_o}{R-y_o} - arctan\frac{x_c}{R-y_c}$.

10: Izračunamo čase do trka: $t_{TTC} = \frac{\phi_c}{\omega},\ t_{TTC} \geq 0$.

11: Poiščemo najmanjšo vrednost t_{TTC}: $TTC_{min} = \text{argmin}(t_{TTC})$.

Če smo v postopku detekcije trka zaznali nevarnost trka, lahko sprožimo akcijo, ki definira obnašanje robota v tovrstnih situacijah. Cilj sistema za preprečevanje trkov je proporcionalno zmanjšanje linearne hitrosti v in krožne hitrosti ω, pri pogoju da ne spremenimo predvidene trajektorije gibanja robota. To lahko storimo tako, da izpolnimo pogoj ohranitve radija trajektorije R:

$$R = \frac{v}{\omega} = \frac{\sigma v}{\sigma \omega}, \tag{61}$$

pri čemer hitrosti zmanjšamo za enak faktor σ. Pri računanju regulacijskega parametra σ upoštevamo le najmanjši čas do trka TTC_{min}, saj je to čas trka z najbližjo oviro.

Algoritem za preprečevanje trkov najprej izračuna pospešek

$$a_r(TTC_{min}, v) = \frac{v}{2TTC_{min}}, \tag{62}$$

kjer je v trenutna linearna hitrost robota. Pospešek a_r ustavi robota v času TTC_{min}, ki potuje s hitrostjo v. Da določimo obnašanje robota, je potrebno definirati spodnjo mejo pospeška $a_{r_{MIN}}$ in zgornjo mejo pospeška $a_{r_{MAX}}$. Če je vrednost pospeška a_r manjša od spodnje meje, potem linearne in krožne hitrosti robota ne spreminjamo. Nasprotno pa vrednost zgornje meje pospeška definira največjo vrednost pojemka. Če je vrednost pospeška a_r večja od zgornje vrednosti, se mora robot ustaviti. Vrednosti pospeška a_r nad zgornjo mejo nakazujejo na to, da se robot ne bo mogel ustaviti v času TTC_{min}. Obnašanje robota glede na vrednosti a_r je opisano v nizu pravil

$$a_r^v = \begin{cases} 0, & a_r(TTC_{min}, v) < a_{r_{MIN}} \\ a_{r_{MAX}}, & a_r(TTC_{min}, v) > a_{r_{MAX}} \\ a_r(TTC_{min}, v), & drugače \end{cases} \tag{63}$$

V vsaki iteraciji algoritma računamo vrednost pospeška a_r^v, ki ga uporabimo za izračun faktorja zmanjšanja

$$\sigma = \frac{v - a_r^v \Delta T}{v}, \tag{64}$$

pri čemer ΔT pomeni časovni interval v katerem periodično izvajamo algoritem za preprečevanje trkov. Faktor zmanjšanja uporabimo pri zmanjšanju linearne in krožne hitrosti robota kot sledi

$$v_{zm} = \sigma v, \quad \omega_{zm} = \sigma \omega \tag{65}$$

V algoritmu 8 je opisan algoritem vodenja robota za preprečevanje trkov.

Algoritem 8 Psevdokoda algoritma za preprečevanje trkov robota pri zavijanju
1: Izberemo časovni interval ΔT.
2: Izberemo spodnjo in zgornjo mejo pogreška ($a_{r_{MIN}}$, $a_{r_{MAX}}$).
3: Izračunamo pospešek a_r: $a_r(TTC_{min}, v) = \frac{v}{2TTC_{min}}$.
4: Če je $a_r < a_{r_{MIN}}$,
5: nastavimo pospešek na predpisano vrednost: $a_r^v = 0$.
6: Če je $a_r > a_{r_{MAX}}$,
7: nastavimo pospešek na zgornjo mejo: $a_r^v = a_{r_{MAX}}$.
8: Če je $a_{r_{MIN}} < a_r < a_{r_{MAX}}$,
9: ne spreminjamo pospeška: $a_r^v = a_r$.
10: Izračunamo faktor zmanjšanja: $\sigma = \frac{v - a_r^v \Delta T}{v}$.
11: Izračunamo in nastavimo zmanjšane hitrosti robota: $v_{zm} = \sigma v, \quad \omega_{zm} = \sigma \omega$.

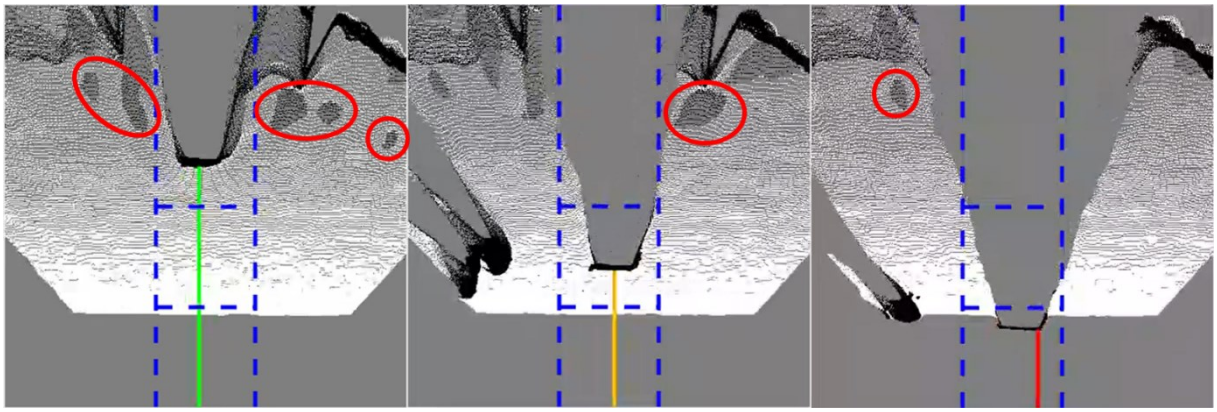
4.3 Rezultati delovanja

V tem poglavju so predstavljeni rezultati delovanja sistemov za preprečevanje trkov pri vožnji naravnost in pri zavijanju. Na sliki 82 je prikazan posnetek prizora, v katerem smo preizkusili delovanje sistema pri vožnji naravnost.

Slika 82. Prizor posnet z levo kamero (levo) in z desno kamero (desno) pri vožnji naravnost

Slika 83 prikazuje obnašanje robota, ko vozi naravnost. V tem primeru nismo uporabljali funkcij postprocesiranja oz. naknadne obdelave slike. Pred robotom so definirana

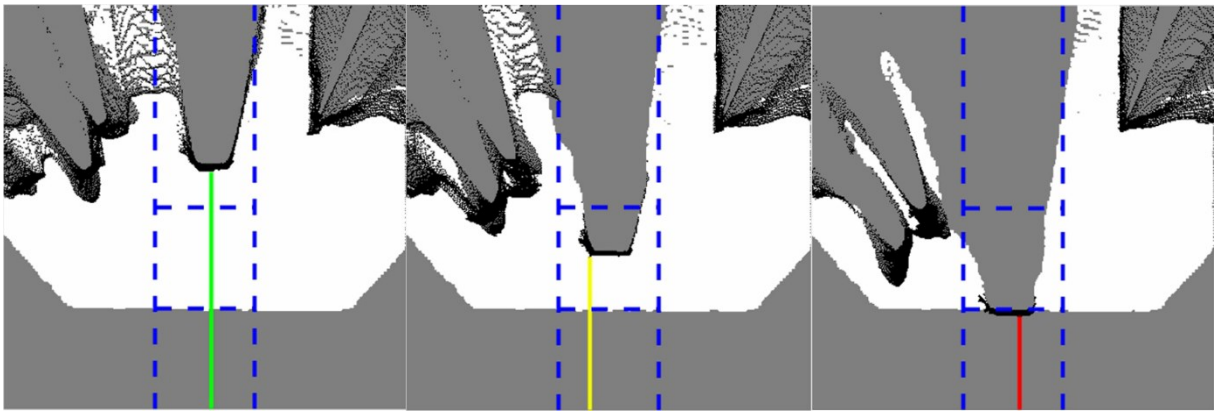
tri območja, ki določajo obnašanje robota. V vsaki iteraciji računamo razdaljo od robota do najbližje točke, ki predstavlja oviro na poti. Zelena črta predstavlja razdaljo do ovire, ki se nahaja v najbolj oddaljenem območju od robota. V tem primeru ovira še ne predstavlja nevarnosti. Ko se približujemo oviri, ki se nahaja v vmesnem območju, robot proporcionalno zmanjšuje hitrost. Na sliki 83 desno je prikazana rdeča črta, ki predstavlja kritično razdaljo do ovire, pri kateri se mora robot ustaviti.



Slika 83. Vožnja naravnost

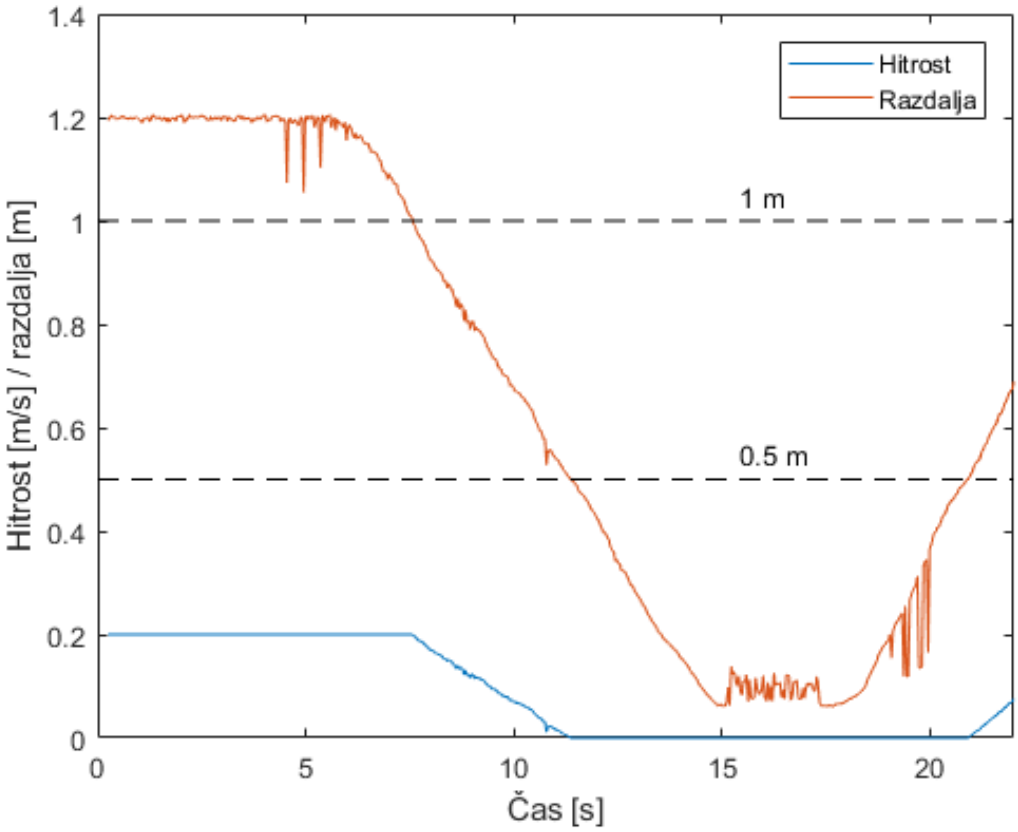
Na lokalnem zemljevidu robota (slika 83) so z rdečimi krogi in elipsami označeni nekateri predeli oz. »črni madeži« lokalnega zemljevida, ki naj bi predstavljali ovire. V tem primeru je prišlo do napačnega izračuna oz. segmentacije okolja, saj so se na teh območjih nahajala le tla. Do napačnega izračuna je verjetno prišlo zaradi zahtevnih svetlobnih pogojev, kjer so prisotni številni odsevi oz. odboji v tleh in izrazite sence ter spremembe intenzitete svetlobe. Na točnost izračuna lahko vpliva tudi varianca meritev, ki z razdaljo narašča. V opisanem algoritmu gradnje lokalnega zemljevida variance ne spreminjamo, kar lahko vpliva na točnost segmentacije zemljevida. Varianco smo določili eksperimentalno glede na velikost opazovanega območja pred robotom.

Na sliki 84 so prikazani izseki lokalnega zemljevida robota, na katerem smo uporabili funkcije postprocesiranja. V tem primeru smo preizkusili delovanje sistema v bolj ugodnih svetlobnih pogojih, kjer opazimo skoraj popolmo odsotnost t. i. črnih madežev. S funkcijami postprocesiranja ne odstranimo črnih madežev, temveč odstarnimo sivinske vrzeli oz. neznane predele, kjer pričakujemo tla.



Slika 84. Vožnja naravnost s postprocesiranjem

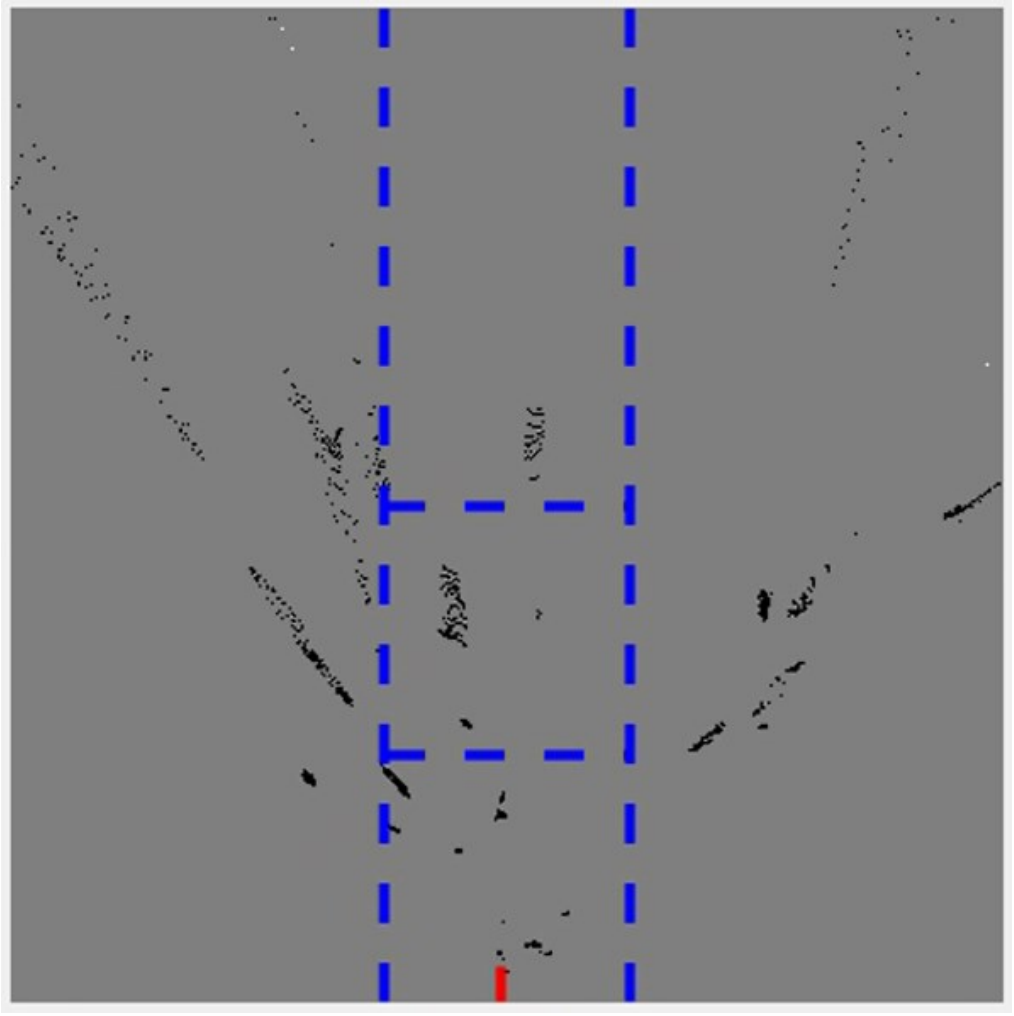
Na sliki 85 je prikazan časovni potek hitrosti robota in razdalje do ovire za primer s slike 83. Robot na začetku vozi z maksimalno hitrostjo, dokler se ne približa oviri, ki se nahaja na razdalji enega metra. Pri razdaljah do ovir, ki so manjše od enega metra, robot proporcionalno zmanjšuje hitrost. Ko se približamo oviri na razdalji 0,5 m ali manj, se robot ustavi. Izbira vrednosti razdalje pri kateri se robot ustavi, je odvisna od lege oz. postavitve kamere na robotu. Paziti moram na tiste ovire, ki se v bližini robota ne nahajajo v vidnem polju kamere. V tem primeru ovire ne moremo zaznati in ne moremo pravilno ukrepati.



Slika 85. Časovni potek hitrosti robota in razdalje do ovire pri vožnji naravnost

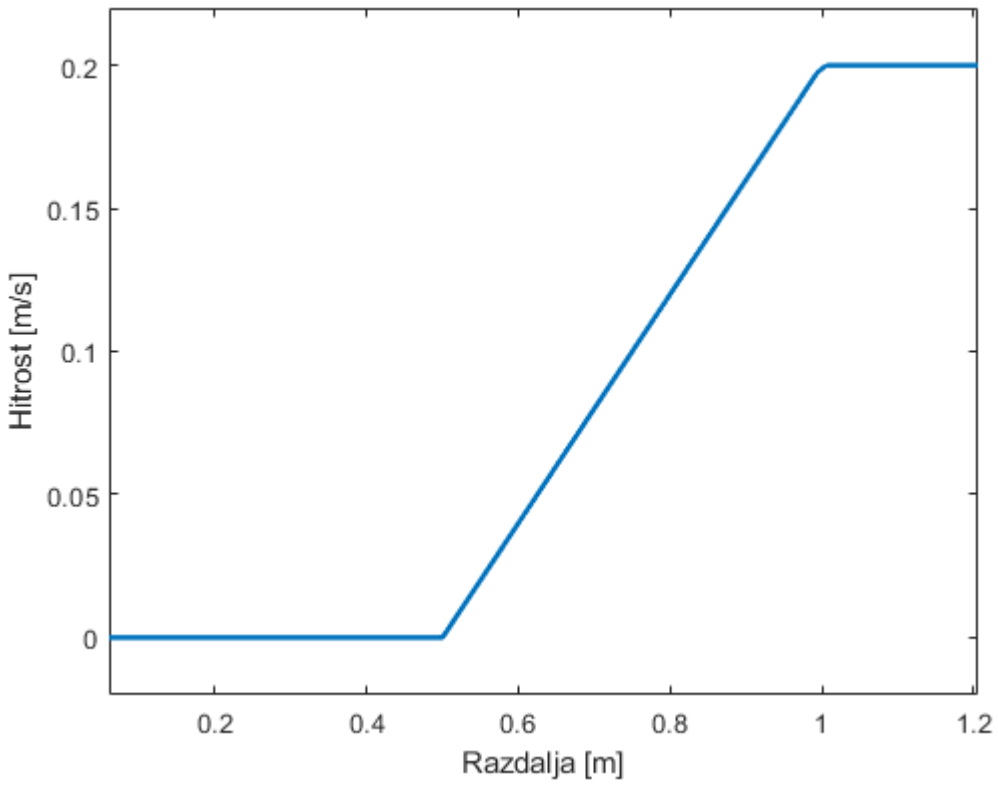
Na sliki 85 opazimo tudi skoke vrednosti razdalj v določenih trenutkih. V intervalu od pet sekund do šest sekund je prisoten šum, ki je lahko posledica različnih odsevov v tleh,

spremembi intenzitete svetlobe, prisotnosti senc, napačnega izračuna globine oz. oblaka točk itn. V intervalu od 15 do 17 sekund se nahaja ovira točno pred robotom oz. globinsko kamero. Za globinsko kamero D435i velja, da pri zajemanju slike z resolucijo 480×270 ne moremo uspešno zaznavati razdalje, ki je manjša od 120 mm. V tem primeru globinska kamera računa napačne rezultate razdalje in posledično opazimo šum na podatkih v tem časovnem intervalu. Ta problem lahko rešimo s programsko analizo meritev. To storimo tako, da sproti beležimo število točk globinske slike, za katere globinska kamera ni uspela izračunati razdalje. Ko število zabeleženih točk preseže določeno vrednost predpostavimo, da senzor ne vrača pravih meritev in zato robota ustavimo. Mejno vrednost števila točk smo določili eksperimentalno. Na sliki 86 je prikazan primer lokalnega zemljevida robota, ko se ovira nahaja točno pred robotom oz. globinsko kamero.



Slika 86. Lokalni zemljevid robota, ko je ovira pred globinsko kamero

Na sliki 87 je prikazan potek hitrosti robota v odvisnosti od razdalje do najbližje ovire. Hitrost robota se zmanjšuje ali povečuje linearno.

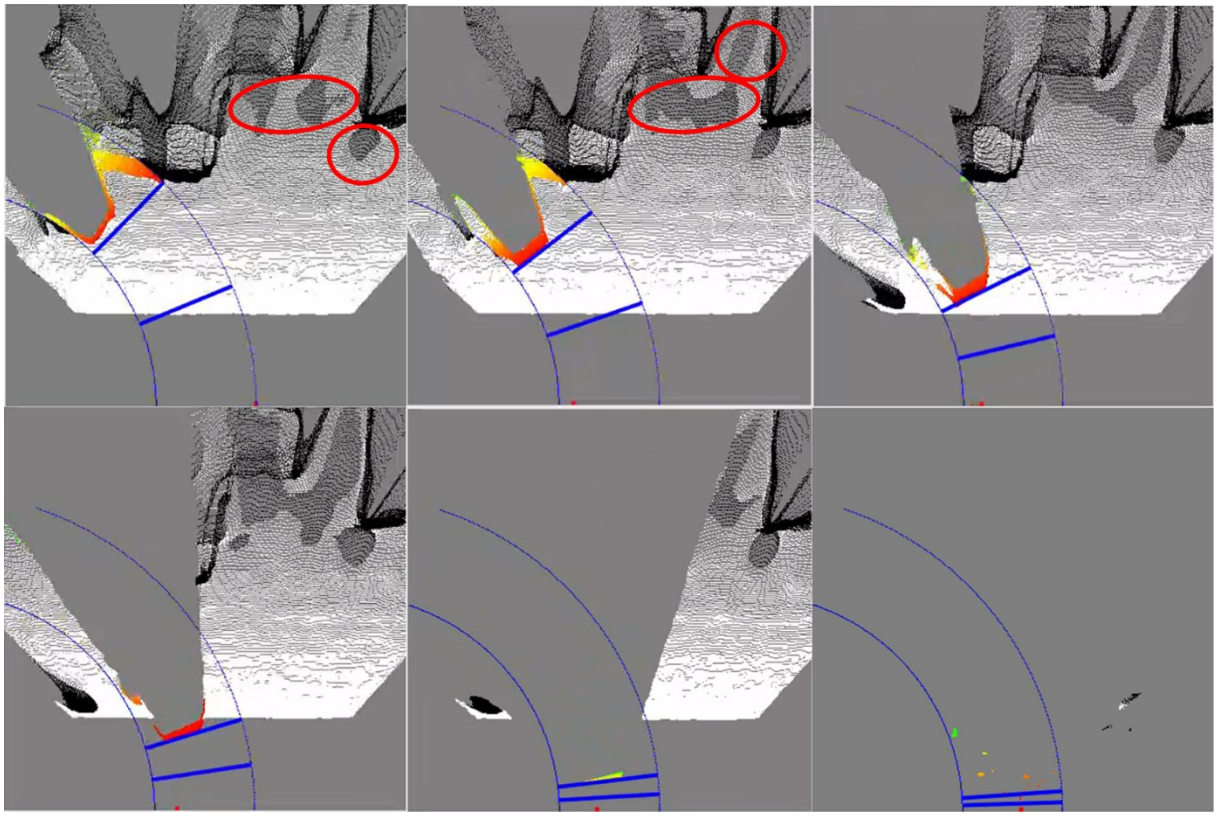


Slika 87. Potek hitrosti robota v odvisnosti od razdalje pri vožnji naravnost

Na sliki 88 je prikazan posnetek prizora, v katerem smo preizkusili delovanje sistema za preprečevanje trkov pri zavijanju.

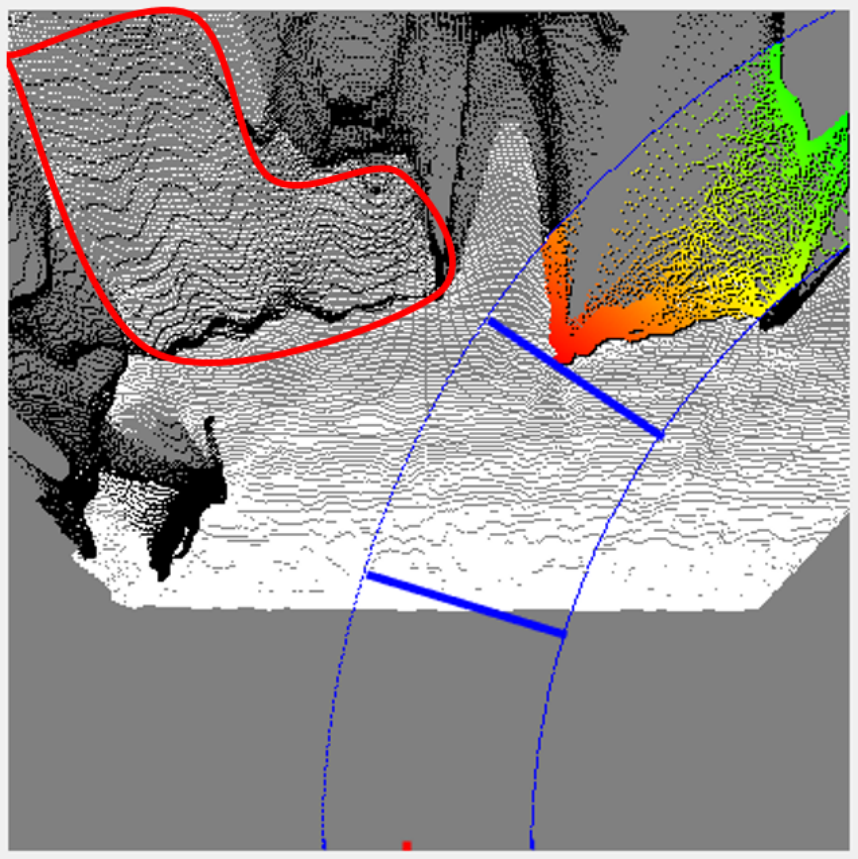
Slika 88. Prizor posnet z levo kamero (levo) in z desno kamero (desno) pri zavijanju

Slika 89 prikazuje lokalni zemljevid robota v postopku zavijanja, kjer nismo vključili funkcije postprocesiranja. Za lažjo predstavitev delovanja sistema na zemljevid izrisujemo trajektorijo gibanja robota, ki je odvisna od linearne in krožne hitrosti robota. Robot zmanjšuje hitrost ali se ustavi glede na prej določene zgornje in spodnje vrednosti pospeška. Na lokalnem zemljevidu robota v postopku zavijanja lahko prav tako opazimo t. i. črne madeže, ki so posledica zahtevnih svetlobnih razmer oz. šuma.



Slika 89. Lokalni zemljevid robota v postopku zavijanja

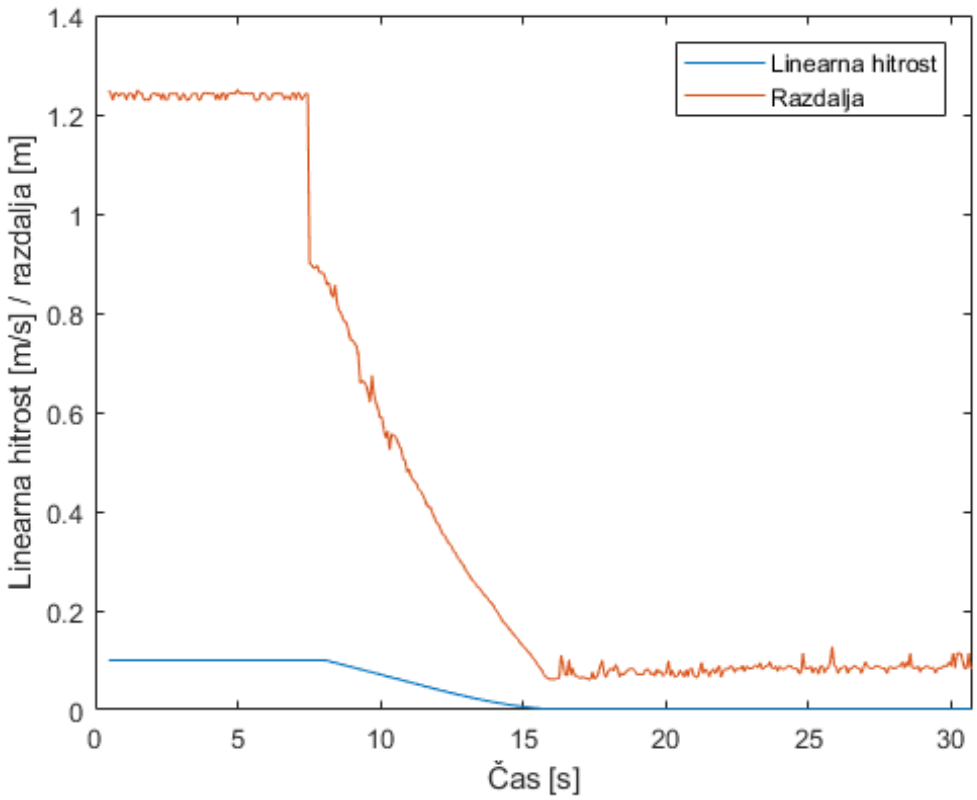
Točke, ki se nahajajo na trajektoriji gibanja robota, predstavljajo ovire. Na sliki 90 je s spreminjanjem barve nazorneje prikazan čas do trka oz. do ovir, ki se povečuje z oddaljenostjo od robota. Rdeča točka v spodnjem delu lokalnega zemljevida predstavlja točko trka na robotu z oviro, ki ima najmanjši čas do trka. Na območju, ki je označeno z rdečo barvo (slika 90), se nahaja miza. Ker smo predvidevali, da je robot višji od mize, se na tem območju nahajajo črne kot tudi bele točke.



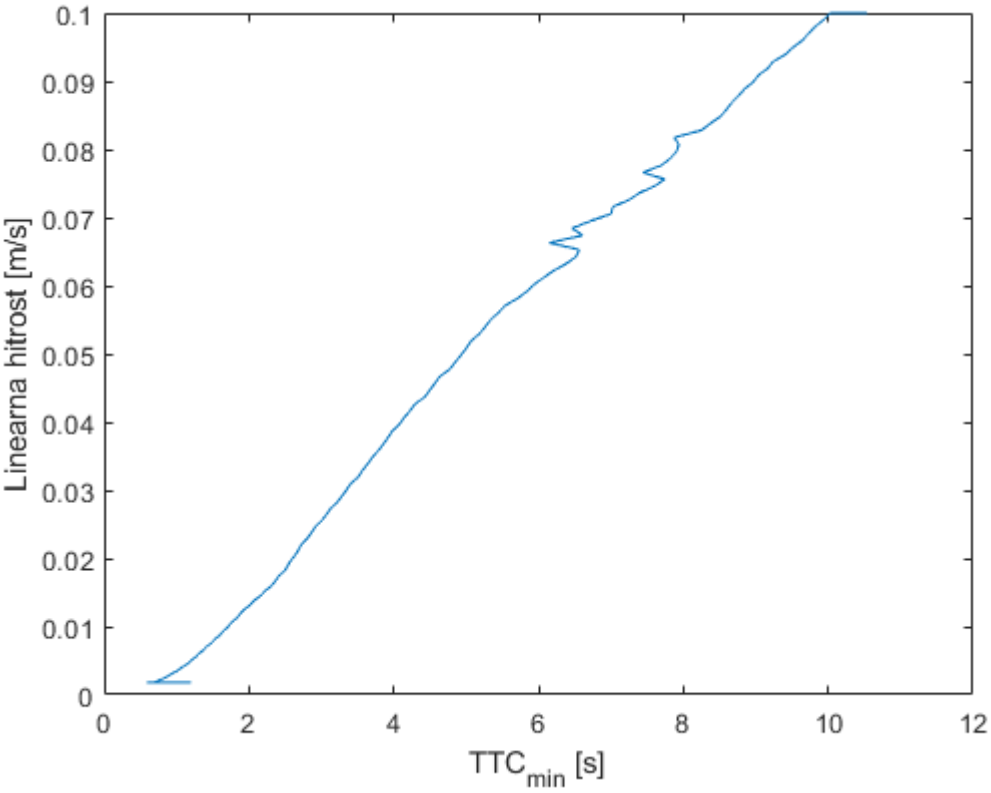
Slika 90. Čas do trka, ki se povečuje z oddaljenostjo

Na sliki 91 je prikazan časovni potek razdalje do najbližje ovire in linearne hitrosti robota za primer s slike 89. Iz poteka hitrosti je ravidno, da se robot ustavi pred oviro in uspešno prepeči trk. Tudi v tem primeru smo vključili varnostni ukrep, ki robota ustavi, ko zaznamo določeno število neuspešno izračunanih globinskih točk.

Na sliki 92 je prikazan potek linearne hitrosti v odvisnosti od časa do trka TTC_{min}. Linearna hitrost se zmanjšuje proporcionalno, vendar so opazna odstopanja zaradi šuma.

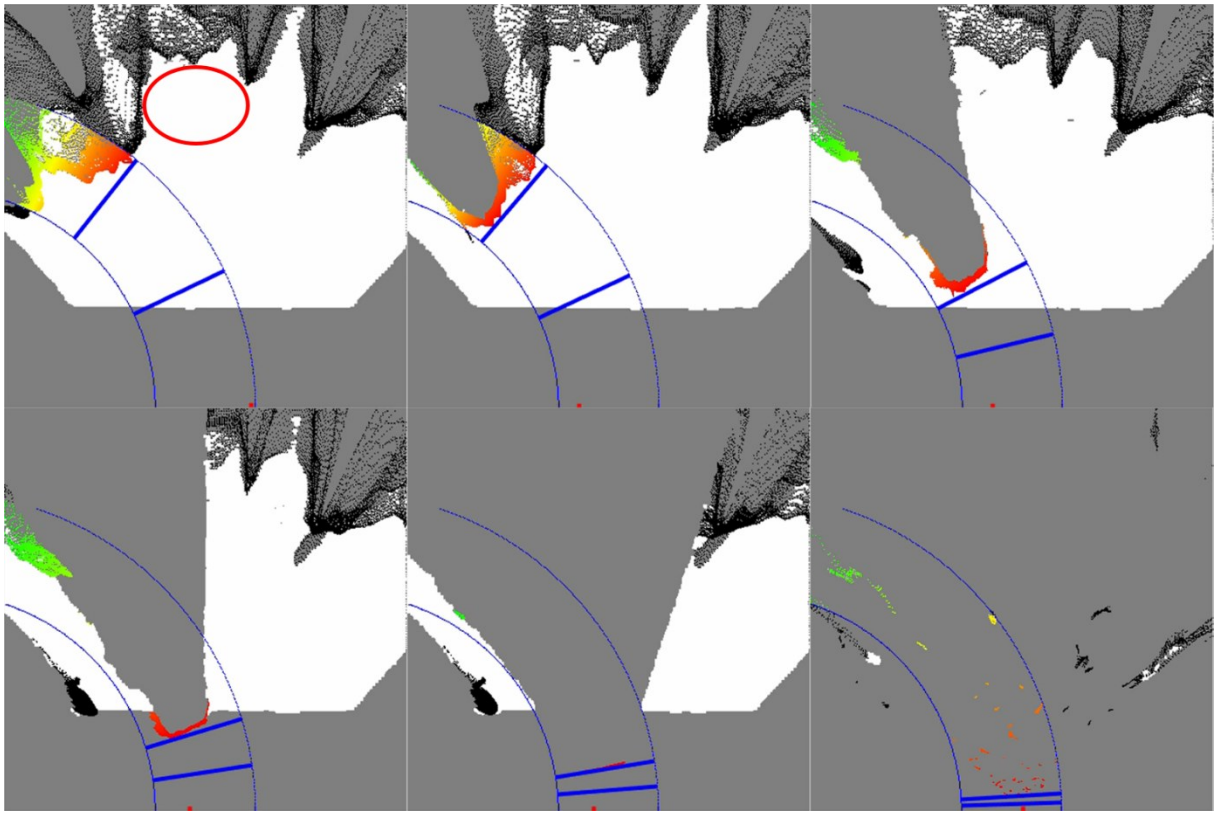


Slika 91. Časovni potek razdalje do najbližje ovire in linearne hitrosti robota



Slika 92. Potek linearne hitrosti v odvisnosti od časa do trka

Na sliki 93 so prikazani izseki lokalnega zemljevida robota v postopku preprečevanja trka pri zavijanju, kjer smo uporabili funkcije postprocesiranja. Na območju, ki je označeno z rdečo barvo, se nahaja miza. V tem primeru smo upoštevali višino robota, ki je nižji od mize, zato je območje pod mizo prosto območje.



Slika 93. Lokalni zemljevid robota z vključenim postprocesiranjem v postopku zavijanja

5 Zaključek

V magistrskem delu smo se lotili izvedbe mobilnega sistema za preprečevanje trkov, ki za interakcijo z okoljem uporablja in obdeluje informacije s stereo kamere. Na mobilnem sistemu je bila nameščena globinsko kamero, ki na izhodu daje globinsko sliko. Na podlagi globinske slike smo rekonstruirali tridimenzionalno predstavitev okolja oz. oblak točk za namen segmentacije okolja oz. detekcije ovir. Da bi mobilnemu sistemu oz. robotu omogočili varno navigacijo v okolju, smo iz tridimenzionalne informacije zgradili lokalni dvodimenzionalni zemljevid robota. Segmentacijo okolja smo definirali kot delitev prostora na ovire, tla in neznana območja. Na območjih oz. razdaljah, kjer kamera snema pod nizkim kotom, pridobimo osiromašeno predstavitev o okolju v primeru izračunanega oblaka točk. Posledično je na dobljenem lokalnem zemljevidu robota veliko neznanih predelov. V ta namen smo na zemljevidu robota uporabili funkcije postprocesiranja. Tako smo zapolnili predele prostora, kjer smo pričakovali tla.

Tekom razvoja sistema smo opazili, da zahtevni svetlobni pogoji vplivajo na kvaliteto izračunanega zemljevida. Ugotovili smo, da prisotnost odsevov oz. odbojev, senc in spremembe intenzitete svetlobe pripomorejo k poslabšanju rezultatov. Na področju strojnega vida predstavlja uspešno iskanje disparitete velik izziv in je še vedno tema mnogih raziskav. Disparitetna slika je pri stereo vidu izhodišče za rekonstrukcijo tridimenzionalne predstavitve okolja. Ker celoten postopek obdelave slik leve in desne kamere poteka na globinski kameri, posledično nismo imeli vpogleda v metodo in uspešnost iskanja disparitetne slike. Med dejavnike, ki vplivajo na točnost in kvaliteto izračunanega zemljevida, lahko uvrstimo tudi metodo iskanja globinske slike, ki jo uporablja globinska kamera.

V tem delu smo prikazali delovanje algoritmov za preprečevanje trkov mobilnega sistema. Implementirani algoritmi so delujoči, vendar je njihova uspešnost odvisna od kvalitete lokalnega zemljevida robota. Sistem za preprečevanje trkov lahko uporabimo kot pomožni podsistem, ki deluje pod višjenivojskimi sistemi za načrtovanje poti. Tako lahko omogočimo varno navigacijo robota v okolju, kjer vedno aktivni podsistem posreduje le v primeru nevarnosti trka.

Literatura

[1] G. Klančar, A. Zdešar, S. Blažič, in I. Škrjanc, *Wheeled mobile robotics: from fundamentals towards autonomous systems*. Oxford; Cambridge (MA): Butterworth-Heinemann, 2017.

[2] G. Klančar, *Avtonomni mobilni sistemi*. Ljubljana: Založba FE, 2014.

[3] „AGV rešitve“. [Na spletu]. Dostopno: https://www.tpv.si/si/agv-resitve/. [Dostopano: 13-avg-2019].

[4] „Sensor Technology“, 2019. [Na spletu]. Dostopno: https://www.syntouchinc.com/en/sensor-technology/. [Dostopano: 16-jul-2019].

[5] „Tactile sensor“. [Na spletu]. Dostopno: https://en.wikipedia.org/wiki/Tactile_sensor#cite_note-A4-2. [Dostopano: 15-avg-2019].

[6] C. Fox, M. Evans, M. Pearson, in T. Prescott, „Tactile SLAM with a biomimetic whiskered robot“, *Proc. - IEEE Int. Conf. Robot. Autom.*, str. 4925–4930, 2012.

[7] R. S. Dahiya in M. Valle, *Robotic Tactile Sensing: Technologies and System*. Berlin: Springer Science+Business Media, 2013.

[8] A. Carullo, M. Parvis, in S. Member, „An Ultrasonic Sensor for Distance Measurement in“, *Sensors (Peterborough, NH)*, let. 1, št. 2, str. 143–147, 2001.

[9] W. Peter in B. Wim, „Ultrasonic Potential Field Sensor for Obstacle Avoidance“, *IEEE Trans. Robot. Autom.*, let. 15, št. 4, str. 774–779, 1999.

[10] C. Wolff, „Radar Basics“, *radartutorial.eu*. [Na spletu]. Dostopno: http://www.radartutorial.eu/index.en.html. [Dostopano: 01-sep-2019].

[11] C. Wolff, „Range or distance measurement“, *radartutorial.eu*. [Na spletu]. Dostopno: http://www.radartutorial.eu/01.basics/Distance-determination.en.html. [Dostopano: 28-jul-2019].

[12] „Radar Sensor“, *ScienceDirect*. [Na spletu]. Dostopno: https://www.sciencedirect.com/topics/engineering/radar-sensor. [Dostopano: 22-jul-2019].

[13] „IR LED | Infrared LED | Infrared Sensor“, *electronicsforu.com*, 2019. [Na spletu]. Dostopno: https://electronicsforu.com/resources/learn-electronics/ir-led-infrared-sensor-basics. [Dostopano: 01-sep-2019].

[14] „How does an IR sensor work?“, *Quora*. [Na spletu]. Dostopno: https://www.quora.com/How-does-an-IR-sensor-work. [Dostopano: 01-sep-2019].

[15] T. Sugimoto in H. Tomooka, „Passive-type infrared sensor system for detecting human body“, US 5,703,368, 1997.

[16] M. Moghavvemi in L. C. Seng, „Pyroelectric infrared sensor for intruder detection“, *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON*, let. D, str. 656-659 Vol. 4, 2004.

[17] E. M. Gorostiza, J. L. L. Galilea, F. J. M. Meca, D. S. Monzú, F. E. Zapata, in L. P. Puerto, „Infrared sensor system for mobile-robot positioning in intelligent spaces“, *Sensors*, let. 11, št. 5, str. 5416–5438, 2011.

[18] A. Shetty, „Infrared Sensor-How it Works. Types; Applications, Advantage & Disadvantage“, *Electricalfundablog.com*. [Na spletu]. Dostopno: https://electricalfundablog.com/infrared-sensor/#Advantages_of_Infrared_Sensor. [Dostopano: 01-sep-2019].

[19] J. Carter *idr.*, „Lidar 101 : An Introduction to Lidar Technology, Data, and Applications“, *National Oceanic and Atmospheric Administration (NOAA) Coastal Services Center*. Charleston, str. 1–72, 2012.

[20] T. Agarwal, „All You Know About LIDAR Systems and Applications“, *ElProCus*. [Na spletu]. Dostopno: https://www.elprocus.com/lidar-light-detection-and-ranging-working-application/. [Dostopano: 28-jun-2019].

[21] „Lidar“, *Wikipedia*, 2019. [Na spletu]. Dostopno: https://en.wikipedia.org/wiki/Lidar. [Dostopano: 01-sep-2019].

[22] R. Behringer *idr.*, „The DARPA Grand Challenge - Development of an Autonomous Vehicle“, *IEEE Intell. Veh. Symp.*, str. 226–231, 2004.

[23] S. Hwang, N. Kim, Y. Choi, S. Lee, in I. S. Kweon, „Fast multiple objects detection and tracking fusing color camera and 3D LIDAR for intelligent vehicles“, *2016 13th Int. Conf. Ubiquitous Robot. Ambient Intell. URAI 2016*, str. 234–239, 2016.

[24] A. Blažic *idr.*, „Autonomous Landing System: Safe Landing Zone Identification“, *SNE Simul. Notes Eur.*, let. 28, št. 4, str. 165–170, 2018.

[25] P. Moghadam, W. S. Wijesoma, in D. J. Feng, „Improving path planning and mapping based on stereo vision and lidar“, *2008 10th Int. Conf. Control. Autom. Robot. Vision, ICARCV 2008*, št. December, str. 384–389, 2008.

[26] C. W. Bater, J. C. White, M. A. Wulder, N. C. Coops, in T. Hilker, „The role of LiDAR in sustainable forest management“, *For. Chron.*, let. 84, št. 6, str. 807–826, 2008.

[27] „Advantages and Disadvantages of LiDAR“, *LIDAR and RADAR Information*, 2018. [Na spletu]. Dostopno: https://lidarradar.com/info/advantages-and-disadvantages-of-lidar. [Dostopano: 01-jul-2019].

[28] „Camera“, *Wikipedia*, 2019. [Na spletu]. Dostopno: https://en.wikipedia.org/wiki/Camera. [Dostopano: 01-sep-2019].

[29] B. Kormann, A. Neve, G. Klinker, in W. Stechele, „Stereo vision based vehicle detection“, *VISAPP*, 2010.

[30] S. Nedevschi *idr.*, „Stere ovision Sensor for Driving Assistance“, *Tech. Univ. Cluj-Napoca, Dep. Comput. Sci.*, 2006.

[31] B. Margrit, H. Esin, in D. Larry S., „Real-time multiple vehicle detection and tracking from a moving vehicle“, *Mach. Vis. Appl.*, let. 12, št. 2, str. 69–83, 2000.

[32] Y. B. Mahdy, K. F. Hussain, in M. A. Abdel-Majid, „Projector Calibration Using Passive Stereo and Triangulation“, *Int. J. Futur. Comput. Commun.*, let. 2, št. 5, str. 385–390, 2013.

[33] „Computer stereo vision“, *Wikipedia*, 2019. [Na spletu]. Dostopno: https://en.wikipedia.org/wiki/Computer_stereo_vision#cite_note-5. [Dostopano: 15-jul-2019].

[34] S. Y. Chen, Y. F. Li, in J. Zhang, „Vision processing for realtime 3-D data acquisition based on coded structured light“, *IEEE Trans. Image Process.*, let. 17, št. 2, str. 167–176, 2008.

[35] W. Jang, C. Je, Y. Seo, in S. W. Lee, „Structured-light stereo: Comparative analysis and integration of structured-light and active stereo for measuring dynamic shape“, *Opt. Lasers Eng.*, let. 51, št. 11, str. 1255–1264, 2013.

[36] J. L. Jones, „Robot obstacle detection system“, US 6,594,844 B2, 2003.

[37] J. L. Jones in R. M. Phillip, „Method and system for multi-mode coverage for an autonomous robot“, US 7,173,391 B2, 2007.

[38] J. Layton, „How Robotic Vacuums Work“, *howstuffworks.com*, 2005. [Na spletu]. Dostopno: https://electronics.howstuffworks.com/gadgets/home/robotic-vacuum2.htm. [Dostopano: 01-sep-2019].

[39] G. W. Landry, D. A. Cohen, in D. Ozick, „Debris sensor for cleaning apparatus“, US 6,956,348 B2, 2005.

[40] M. Chen, Z. Cai, in Y. Wang, „A method for mobile robot obstacle avoidance based on stereo vision“, *IEEE Int. Conf. Ind. Informatics*, št. 1, str. 94–98, 2012.

[41] G. Bradski in K. Adrian, *Learning OpenCV*, First edit. Sebastopol, CA: O'Reilly, 2008.

[42] „Future of driving“, *tesla.com*, 2019. [Na spletu]. Dostopno: https://www.tesla.com/autopilot?redirect=no. [Dostopano: 01-sep-2019].

[43] M. R. Endsley, „Autonomous Driving Systems: A Preliminary Naturalistic Study of the Tesla Model S“, *J. Cogn. Eng. Decis. Mak.*, let. 11, št. 3, str. 225–238, 2017.

[44] „Waymo 360° Experience: A Fully Self-Driving Journey“, *Youtube*, 2019. [Na spletu]. Dostopno: https://www.youtube.com/watch?time_continue=26&v=B8R148hFxPw. [Dostopano: 01-sep-2019].

[45] „Technology“, *Waymo*, 2019. [Na spletu]. Dostopno: https://waymo.com/tech/. [Dostopano: 01-sep-2019].

[46] „How does R1 work?“, *Skydio Inc.*, 2019. [Na spletu]. Dostopno: https://support.skydio.com/hc/en-us/articles/360000593334-How-does-R1-work-. [Dostopano: 15-avg-2019].

[47] „Using Skills within the Skydio app“, *Skydio Inc.*, 2019. [Na spletu]. Dostopno: https://support.skydio.com/hc/en-us/articles/360001597514. [Dostopano: 15-avg-2019].

[48] Skydio Incorporation, „Skydio Safety and Operating Guide: Read before flight“. str. 1–2, 2018.

[49] J. Perš, „Computer Vision 03b - Lens distortion“. Fakulteta za elektrotehniko, Laboratorij za strojno inteligenco, Ljubljana, str. 2–21.

[50] F. Devernay in O. Faugeras, „Straight lines have to be straight : automatic calibration and removal of distortion from scenes of structured enviroments“, *Mach. Vis. Appl.*, let. 13, št. 1, str. 14–24, 2008.

[51] C. Hughes, P. Denny, E. Jones, in M. Glavin, „Accuracy of fish-eye lens models“, *Appl. Opt.*, let. 49, št. 17, str. 3338–3347, 2010.

[52] A. Grunnet-Jepsen, J. N. Sweetser, in J. Woodfill, „Stereo depth cameras for mobile phones“, *Intel RealSense Technology*. [Na spletu]. Dostopno: https://dev.intelrealsense.com/docs/stereo-depth-cameras-for-phones. [Dostopano: 26-avg-2019].

[53] „Intel RealSens Depth Camera D435i“, *Technology Intel RealSense*. [Na spletu]. Dostopno: https://www.intelrealsense.com/depth-camera-d435i/. [Dostopano: 01-sep-2019].

[54] „Calibration Tools User Guide for Intel RealSense D400 Series“, *Intel RealSense Technology*. [Na spletu]. Dostopno: https://dev.intelrealsense.com/docs/intel-realsensetm-d400-series-calibration-tools-user-guide. [Dostopano: 01-sep-2019].

[55] D. C. Brown, „Close-Range Camera Calibration“, *Photogramm. Eng.*, let. 37, št. 8, str. 855–866, 1971.

[56] J. Perš, „Computer Vision 13 - Stereo matching“. Fakulteta za elektrotehniko, Laboratorij za strojno inteligenco, Ljubljana, str. 63–85.

[57] „disparity“, *Mathworks - Documentation*. [Na spletu]. Dostopno: https://www.mathworks.com/help/vision/ref/disparity.html. [Dostopano: 26-avg-2019].

[58] H. Hirschmu, „Stereo Processing by Semiglobal Matching and Mutual Information“, *IEEE Trans. Pattern Anal. Mach. Intell.*, let. 30, št. 2, str. 328–341, 2008.

[59] D. G. Lowe, „Distinctive image features from scale-invariant keypoints“, *Int. J. Comput. Vis.*, let. 60, št. 2, str. 91–110, 2004.

[60] H. Bay, A. Ess, T. Tuytelaars, in L. Van Gool, „Speeded-Up Robust Features (SURF)“, *Comput. Vis. Image Underst.*, let. 110, št. 3, str. 346–359, 2008.

[61] J. Matas, O. Chum, M. Urban, in T. Pajdla, „Robust wide-baseline stereo from maximally stable extremal regions“, *Image Vis. Comput.*, let. 22, št. 10 SPEC. ISS., str. 761–767, 2004.

[62] E. Rosten in T. Drummond, „Fusing Points and Lines for High Performance Tracking (FAST)“, *Proc. IEEE Int. Conf. Comput. Vis.*, let. 2, str. 1508–1511, 2005.

[63] E. Mair, G. D. Hager, D. Burschka, M. Suppa, in G. Hirzinger, „Adaptive and generic corner detection based on the accelerated segment test“, v *Computer Vision –ECCV 2010*, Springer-Verlag, Berlin, Heidelberg, 2010, str. 183–196.

[64] „The OpenCV reference manual: release 3.0.0“, 2014. [Na spletu]. Dostopno: https://docs.opencv.org/3.0-beta/opencv2refman.pdf.

[65] A. Grunnet-Jepsen, J. N. Sweetser, T. Khuoung, D. Tong, in J. Woodfill, „Subpixel Linearity Improvement for Intel® RealSense™ Depth Camera D400 Series“, *Intel RealSense Technology*. [Na spletu]. Dostopno: https://dev.intelrealsense.com/docs/white-paper-subpixel-linearity-improvement-for-intel-realsense-depth-cameras. [Dostopano: 13-avg-2019].

[66] „Intel® RealSenseTM D400 Series Product Family: Datasheet“, *Intel Corporation*, 2019. [Na spletu]. Dostopno: https://www.intelrealsense.com/wp-content/uploads/2019/07/Intel-RealSense-D400-Series-Datasheet-Jun-2019.pdf?_ga=2.120327624.1187815166.1565094025-949185072.1561497063. [Dostopano: 13-avg-2019].

[67] A. Grunnet-Jepsen in D. Tong, „Depth Post-Processing for Intel RealSense D400 Depth Cameras“, *Intel RealSense Technology*. [Na spletu]. Dostopno: https://dev.intelrealsense.com/docs/depth-post-processing. [Dostopano: 01-sep-2019].

[68] A. Grunnet-Jepsen, J. N. Sweetser, in J. Woodfill, „Tuning depth cameras for best performance“, *Intel RealSense Technology*. [Na spletu]. Dostopno: https://dev.intelrealsense.com/docs/tuning-depth-cameras-for-best-performance. [Dostopano: 01-sep-2019].

[69] „Infinite impulse response“, *Wikipedia*. [Na spletu]. Dostopno: https://en.wikipedia.org/wiki/Infinite_impulse_response. [Dostopano: 14-avg-2019].

[70] C. Godard, O. Mac Aodha, in G. J. Brostow, „Unsupervised monocular depth estimation with left-right consistency“, *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, let. 2017-Janua, str. 6602–6611, 2017.

[71] J. Perš, „Computer Vision 09 – Image processing and analysis 2a“. Fakulteta za elektrotehniko, Laboratorij za strojno inteligenco, Ljubljana, str. 28–51.

[72] M. Bošnak in I. Škrjanc, „Efficient time-to-collision estimation for a braking supervision system with LIDAR“, *2017 3rd IEEE Int. Conf. Cybern. CYBCONF 2017 - Proc.*, 2017.

[73] E. W. Weisstein, „Circle-Line Intersection“, *From MathWorld--A Wolfram Web Resource*. [Na spletu]. Dostopno: http://mathworld.wolfram.com/Circle-LineIntersection.html. [Dostopano: 24-avg-2019].