SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. xxxx

**Lokalizacija površinskih vozila korištenjem sustava kamera**

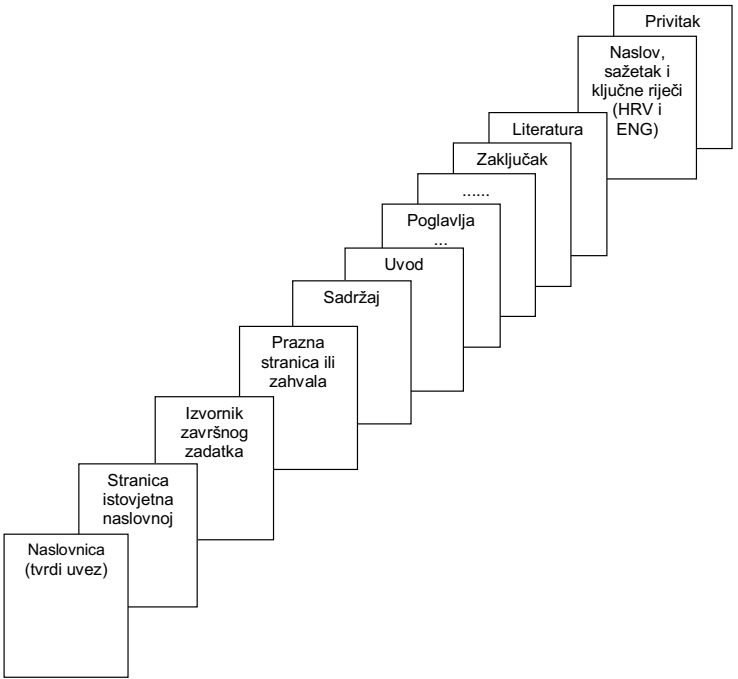
Leonardo Roy Sabolić

Zagreb, lipanj 2025.

Završni rad predaje se u elektroničkoj verziji u pdf obliku.

Student u bazu učitava rad od četvrte stranice (Prazna stranica ili zahvala). Prve tri stranice (naslovnica, stranica istovjetna naslovnici, završni zadatak) bit će automatski povučene iz sustava.

Prilikom oblikovanja rada držite se redoslijeda. Redoslijed stranica okvirno je prikazan na slici.



Student predaje samo pdf verziju rada preko FERweb-a.Ova stranica treba biti:

ILI prazna stranica

ILI stranica sa zahvalom (po želji studenta; zahvala nije obvezna).

Sadržaj

[Uvod 1](#_Toc200375784)

[1. Korišteni alati 3](#_Toc200375785)

[1.1. OpenCv 3](#_Toc200375786)

[1.2. Boost 3](#_Toc200375787)

[1.3. ROS2 4](#_Toc200375788)

[1.4. Arduino 4](#_Toc200375789)

[2. Generalni opis sustava i izmjene 5](#_Toc200375790)

[2.1. Koncept sustava 5](#_Toc200375791)

[2.1.1. Node 5](#_Toc200375792)

[2.1.2. Connector 6](#_Toc200375793)

[2.2. Izmjene sustava 6](#_Toc200375794)

[2.2.1. Izmijenjeni Node-ovi 6](#_Toc200375795)

[3. Sustav za kašnjenje 7](#_Toc200375796)

[3.1. Princip rada 7](#_Toc200375797)

[3.2. Implementacija 7](#_Toc200375798)

[4. Sustav za praćenje 8](#_Toc200375799)

[4.1. Izbor načina za praćenje 8](#_Toc200375800)

[4.2. Princip rada 8](#_Toc200375801)

[4.3. Implementacija 8](#_Toc200375802)

[5. Testiranje sustava 9](#_Toc200375803)

[6. Naslov prvog poglavlja 10](#_Toc200375804)

[6.1. Prvo potpoglavlje 10](#_Toc200375805)

[6.2. Stilovi za tekst, naslove i podnaslove 10](#_Toc200375806)

[6.3. Stilovi za nabrajanje 10](#_Toc200375807)

[6.3.1. Stilovi za nabrajanje s točkama i crticama 11](#_Toc200375808)

[6.4. Slike 12](#_Toc200375809)

[6.5. Tablice 14](#_Toc200375810)

[6.6. Matematički izrazi i formule 15](#_Toc200375811)

[6.7. Programski kôd 15](#_Toc200375812)

[Zaključak 17](#_Toc200375813)

[Literatura 18](#_Toc200375814)

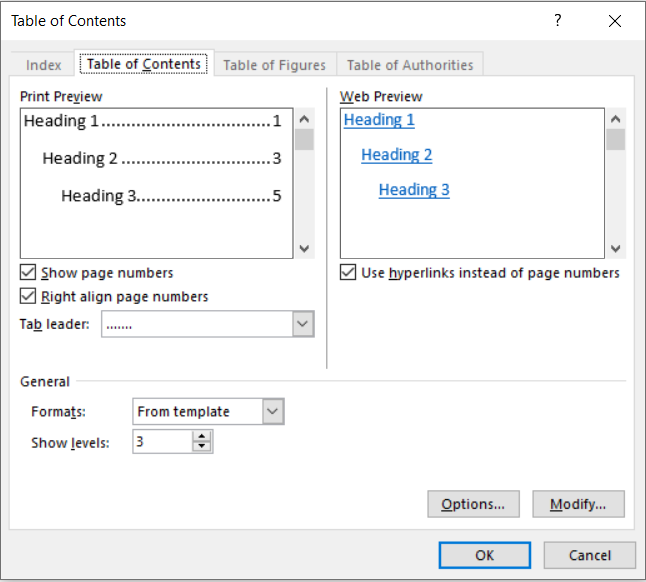
[Sažetak 19](#_Toc200375815)

[Summary 20](#_Toc200375816)

[Skraćenice 21](#_Toc200375817)

[Privitak 22](#_Toc200375818)

Sadržaj se kreira automatski pomoću opcija **References** – **Table of Contents – Insert Table of Contents**. Nakon toga se otvara sljedeći prozor:



Za primjenu jednostavnog stila sadržaja (kao u ovom dokumentu), dovoljno je samo odabrati opciju [OK].

# Uvod

U robotici se često suočavamo s problemom praćenja pozicije nekog objekta, te postoje razne metode određivanja pozicije to mogu bit unutrašnji ili vanjski senzori od kojih se koriste razne metode.

U ovom radu fokusirat ćemo se na vanjski sustav koji koristi kamere. Tu se već susrećemo s dosta problematike, ovisno o sustavu kamera implementacija našeg praćenja može znatno varirati, udaljenost, vidno polje, svjetlosni uvjeti, brzina i rezolucija kamere, i drugi parametri jako utječu na izbor načina praćenja i kompleksnost sustava za praćenje. Glavni problem kojim se ovaj rad bavi je problematika kod izrade takvih sustava na brz i efikasan način, te mogućnost brze iteracije i testiranja raznih takvih sustava. Ovaj rad se nastavlja na rad od ime prezime naslov, koji se bavi dizajnom samog sučelja programa i funkcionalnostima kalibracije kamere na par načina. Glavni cilj je napraviti sustav koji će omogućiti lagano brzo i precizno postavljanje proizvoljnog sustava kamera, te povezivanje tog sustava s drugim robotskim sustavima, time se omogućuje i rad na nepoznatim lokacijama s nepoznatim sustavom kamera ili lako testiranje i kalibriranje pri izmjeni sustava kamera.

Glavnina rada se bavi nadogradnjama i izmjenama sustava iz navedenog rada, povećava se fleksibilnost, dodaju se funkcionalnosti praćenja i sinkronizacije među kamerama, te se testira funkcionalnost cijelog sustava. Sustav bi u konačnici bio korišten za praćenje površinskih vozila ali bi bio primjenjiv i na bilo koji robotski sustav koji koristi neki statičan sustav kamera za navigaciju.

# Korišteni alati

Rad je rađen dijelom na Windows 11 OS-u u C++ programskom jeziku te je za sučelje korištena ImGui biblioteka, više o sučelju je opisano u navedenom radu, neke od glavnih biblioteka korištenih su OpenCv i Boost. Drugi dio je rađen na Ubuntu Linux OS-u na ROS2 platformi. Od hardvera je korišten Arduino Uno, logithec c270 kamere i sustav kamera u LABUST-u.

## OpenCv

Biblioteka koja se koristi za obradu slike i to postiže kroz obradu slike u matričnoj reprezentaciji te je time pogodna i za manipulacije matricama. U radu se koristi za dobivanje slike, filtraciju željenih elemenata na slikama, manipulaciju i dobivanje transformacijskih matrica, te ima ugrađenu podršku za praćenje ArUco markera koji se koriste u nekim dijelovima rada. Zbog svega toga ova biblioteka je idealni izbor jer pokriva puno potrebnih funkcionalnost za izvršavanje ovog rada.

## Boost

Vrlo velika biblioteka s raznim alatima koji bi mogli biti potrebni kod developmenta raznih vrsta aplikacija, neke od najčešće korištenih značajki su pametni pokazivači (*eng. smart pointers*), asinkroni I/O u sklopu Boost.Asio, više drevnost (*eng. multy threading*) u sklopu Boost.Thred, Boost.Filesystem i drugi manje poznati dijelovi biblioteke.

Od ovih funkcionalnosti za ovaj rad je najbitniji Boost.Asio, te se takoder koriste Boost-ovi pametni pokazivači koji su integrirani u C++11 nadalje.

Boost.Asio je biblioteka koja se bavi komunikacijom preko mreže ili bilokakvih ulazno izlaznih jedinica. U radu se koristi pri serijskoj komunikaciji preko USB porta i pri UDP komunikaciji preko mreže.

## ROS2

ROS je platforma koja se koristi u Robotici za laganu komunikaciju raznih sustava potrebnih za upravljanje, percepciju i donošenje odluka u nekom Robotskom sustavu.

U sklopu rada se koristi ROS2 Humble, te se preko njega aplikacija koja se razvija povezuje s ostatkom robotskog sustava, ovaj sustav je pogodan jer se lako preko mreže mogu povezati razni sustavi i izmjenjivati podatke. Ovaj sustav podržava rad u Python ili C++ programskim jezicima te je korišten Python radi njegove jednostavnosti.

## Arduino

Arduino je jednostavan programibilan mikro kontroler koji se koristi uglavnom za prikupljanje senzorskih podataka i upravljanje aktuatorima i drugim sklopovima. U radu se koristi Arduino Uno 8bit-ni mikro kontroler izabran radi niske zahtjevnost zadatka koju izvršava, dodatna prednost je niska cijena. \*

# Generalni opis sustava i izmjene

Sustav je implementiran kao sučelje koje korisniku omogućuje slaganje i spajanje node-ova koji su zasebni procesi za izvršavanje određenih zadatka, te se nadovezivanjem vise takvih node-ova odbija neki željeni efekt. Cijeli sustav je napravljen da radi na sto fleksibilnijem sustavu kamera sto uključuje proizvoljan broj, karakteristike i položaje kamera. \*

## Koncept sustava

Sustav je koncipiran kao Node/Connection sistem, svaki Node ima konektore koji su ili ulazni ili izlazni te svaki konektor prima ili šalje određeni tip poruke te se dodavanjem i spajanjem Node-ova prenose obrađeni podaci medu njima.

### Node

Svaki Node ima istih par funkcija u kodu pomoću kojih se definiraju konektori i njihovi tipovi, u funkcijama getInMessageTypes() i getOutMessageTypes() se definiraju ulazni i izlazni konektori. Funkcije drawNodeWork() i drawNodeParams() se izvršavaju svako iscrtavanje programa i služe za prikaz elemenata Node-a jedan je namjenjen za stalne elemente a jedan za promjenjive koji ovise o nekakvom procesiranju. Funkcija recieve() se pokreće svaki put kada je primljena poruka te se iz primljene poruke ona može po nekom svojstvu odrediti kako dalje procesirati poruku. \*

Opisi Node-ova:

Većina Node-ova je opisano u radu rad.

* NodeSourceStream
  + Služi za spajanje na kamere i definiranje parametara, također služi za intrinzičnu kalibraciju kamere, više o tom je napisano u navedenom radu.
* Node-color-treshold
  + Služi za filtraciju određene boje iz slike, detaljniji opis je u navedenom radu.
* Node-inflate
  + Služi za popravljenje rubova nakon filtracije kako bi se dobila stabilnija filtrirana sliak, detaljnije opisan u navedenom radu.
* Node-background-subtraction
  + Služi za micanje pozadine kako bi se izdvojio objekt, detaljnije opisan u navedenom radu.
* Blob-creator
  + Služi za kreiranje blobova (određeni tip podatka) iz filtrirane slike, detaljnije opisano u navedenom radu.
* Blob-grouper
  + Služi za automatsku ekstrinzičnu kalibraciju pomoću detektiranog filtriranog bloba, detaljnije objašnjeno u navedenom radu.
* Node-manual-extrinsics
  + Služi za ekstrinzičnu kalibraciju putem izbora točaka koje izabire korisnik, i za definiranje nekog proizvoljnog koordinatnog sustava i izračun transformacijske matrice, detaljnije objašnjeno u navedenom radu.
* Node-delay-measurment
  + Služi za mjerenja kasnjenja jedne ili vise kamera pomocu spajanja na vanjski mikrokontroler, detaljniji o radu je opisano u naslov.
* Node-aruco-tracking
  + Sluzi za pracenje ArUco markera u odabranom kordinatnom sustavu, radi na proizvoljnom broju kamera i salje podatke o markerima na zeljenu ip addresu : port, detaljnije opisan rad u naslov.

### Konektor

Konektori su vizualno prikazani na Node-ovima s lijeve i desne strane, s lijeve su ulazni konektori a s desne izlazni, pored svakog konektora piše tip poruke koji prima ili šalje, konektori se vizualno povezuju plavim linijama.

U implementaciji se komunikacija postiže dijeljenim pokazivaćima (*eng. shared pointer*) te se koristi funkcija send() ili sendAll() koje šalju poruku na neki konektor s željenim id-em ili na sve konektore.

Postoji problematika u ovakvoj implementaciji a to je da je dosta teško imati više izlaznih konektora istog tipa jer nema neki način za odrediti koji je koji konektor, moguće je dobiti dodijeljeni id svakog konektora ali on nam ne pomaže pri razlikovanju konektora pri inicijalizaciji node-a, ova situacija se trenutno nigdje ne koristi ali bi bilo korisno imati mogućnost dodijeliti ime konektoru pri inicijalizaciji radi lakšeg dodjeljivanja željenih poruka. Primjer bi bio recimo node za filtriranje slike, umjesto da koristimo 3 takva jedan bi bio dovoljan za filtraciju svih ulaznih kamera te je to trenutno nemoguće, ali bi u budućoj implementaciji to bilo dobro popraviti.

## Izmjene sustava

Sustav je u svojem početnom stanju bio dosta nefleksibilan s nekim stvarima kao što su rezolucija, način kalibracije i imao je neke manje probleme. Zbog toga je za svaki node dodana mogućnost korištenja bilo koje rezolucije, te korištenje bilo koje kalibracijske ploče (crno bijela šahovnica).

### Izmijenjeni Node-ovi

Najviše izmijenjen node je NodeSourceStreme gdje je dodana mogućnost upis veličine kalibracijske ploče, veličine kockica na ploči, dodana je mogućnost izbora rezolucije, te je dodan ulazni parametar tipa int(cijeli broj) koj služi za primanje podatka o kašnjenju kamere. U Node-manual-extrinsics je dodana rezolucija te je to inkorporirano u kalkulacije extrinzičnih matrica, postoji jedan problem koji nastaje zbog logike izbora parova točaka a to je da to sučelje računa s tim da je rezolucija prozora za izbor točaka jednaka rezoluciji kamera, te da je udaljenost među njima 8px. Problem s tim je što nije lagano birati točke na većim rezolucijama bar na monitoru slične rezolucije.

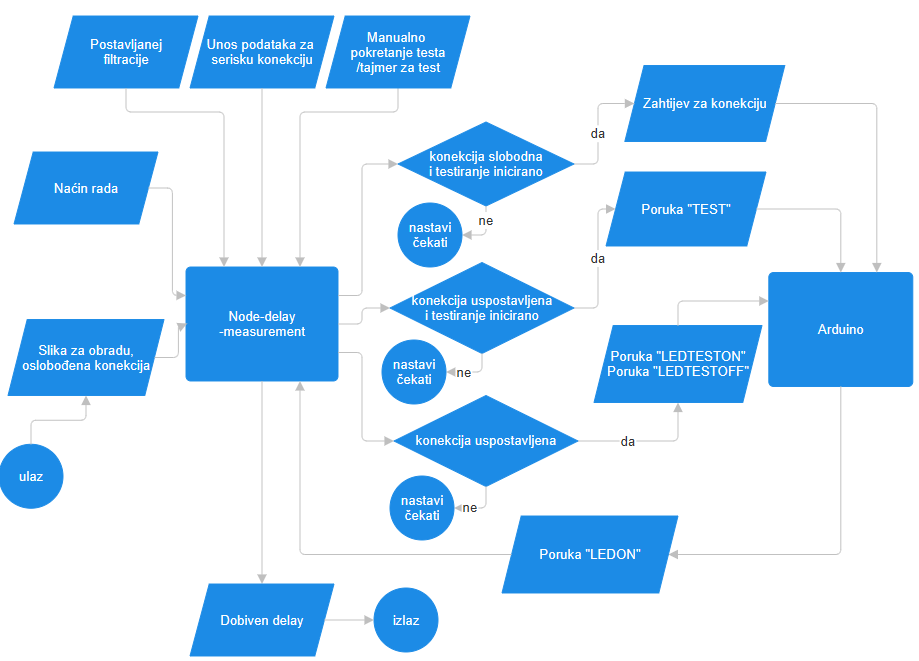
# Sustav za kašnjenje

Sustav za kašnjenje je napravljen kao Node-delay-measurmant, ulazni konektor prima podatak sliku i cijeli broj koji služi za sinkronizaciju testova među više instanci node-a, na izlazu vrača cijeli broj kašnjenje u milisekundama (ms) i filtriranu sliku.

Moguće je testiranje kašnjenja bez sinkronizacije ali se može javiti problem kada jedna od instanci node-a dobi konekciju na Arduino ali se usred inicijalizacije dogodi zahtjev konekcije od drugog te se onda Arduino zamrzne, zato je dodana sinkronizacija a time se dodatno izbjegava zagušenje serijskog kanal s zahtjevima za spajanje. Ovaj problem se ne javlja kod korištenja zasebnog Arduina za svaku instancu node-a.

## Princip rada

Slika .. Dijagram rada

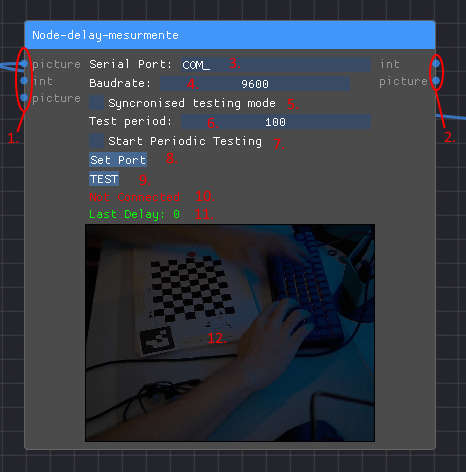


Node prima sliku od kamere, te se pokretanjem testiranja šalje poruka (TEST) vanjskom mikro kontroleru koji povratno pali lampicu i vraća poruku (LEDON), te onda Node prati vrijeme od dobivanja poruke od detekcije željene značajke na slici, te se prosljeđuje dobivena razlika u vremenu na izlazni konektor. Kod više testova s istim mikro kontrolerom se onda dobivena vrijednost prosljeđuje slijedećem Node-u te on onda pokreće testiranje i ponavljaju se već navedeni koraci.

Poruke LEDTESTON i LEDTESTOFF su poruke koje se šalju kod postavljanja filtera za detekciju izabrane značajke te se odvije radi o paljenju i gašenju led diode, ideja je da se filter postavi s upaljenom led diodom te se onda unaprijed vidi kvaliteta filtera i detekcije.

## Implementacija

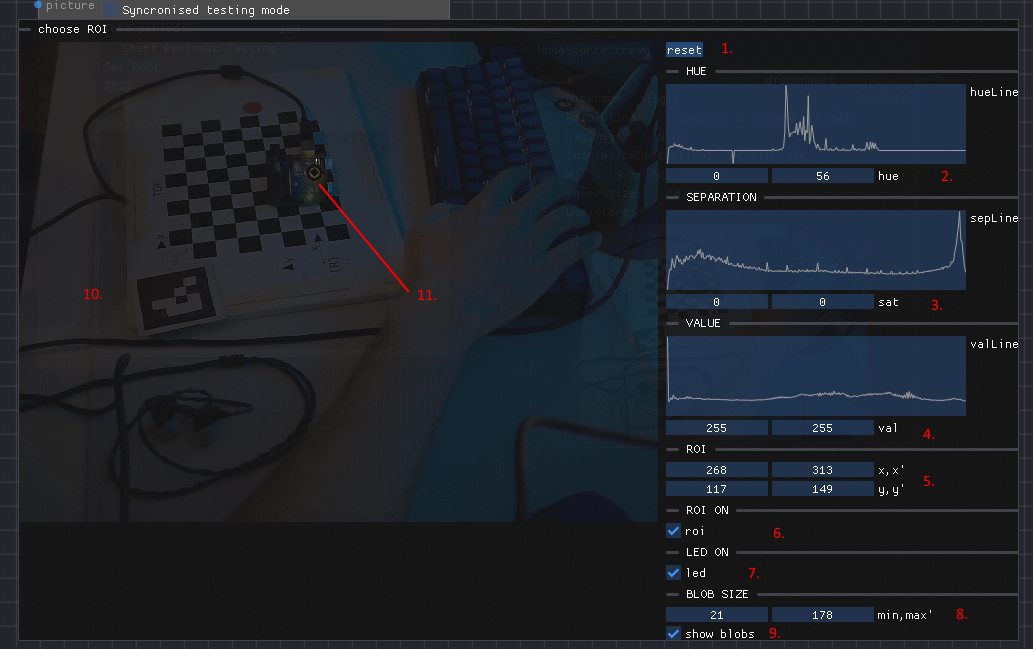
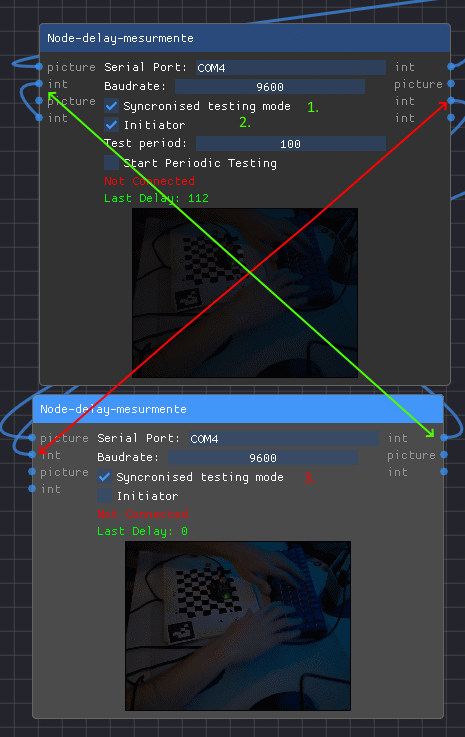
Slika . Primjer Node-delay-measurment



Node prima sliku i int na ulazu (1. ulazni konektori) te se onda slika filtrira (12. prikaz filtrirane slike). Node prima parametre za serijsku komunikaciju (3. i 4.) s Arduinom, te se onda pri pokretanju testa manualno, na tajmeru ili putem drugog Node-a pokuša uspostaviti konekcija s Arduinom (10. indikator konekcije) pri uspješnoj konekciji se tada pokreće testiranje, te se rezultat testa prikazuje (11.) i šalje na izlaz (2.). Na izlazu se također šalje i filtrirana slika. Gumbi (8. i 9.) služe za manualno spajanje i testiranje, a za automatsko se postavlja periodičko testiranje (6. i 7.) s periodom u sekundama ili ako se koristi sinkronizirano testiranje onda se Node postavlja u taj način rada (5.) . Pritiskom na filtrirani prikaz (12.) se otvara sučelje za postavljanje filtera.

Pri korištenju sinkroniziranog načina rad je bitno povezati sve Nodove koji koriste isti Arduino u loop kao što je prikazano s zelenim i crvenim strelicama, još je bitno imati inicijatora (2.) koji će pokretati testiranje periodički, bitno je isto sve node-ove staviti u sinkronizirani način rada (1. i 3.). Ova implementacija nije baš naj ljepša vizualno ali izbjegava nepotrebno gušenje serijskog porta. Bolja solucija bi bila imati jedan node s više ulaza za sliku i više filtera i testova za svaku sliku, te bi tako mogli sve istovremeno testirati, no problem je kod izlaza jer trenutna implementacija nema laki način za određivanje koji izlazni port će se dodijeliti kome.

Slika . Testiranja u sinkroniziranom načinu rada



Slika . Sučelje za filtraciju slike

Za filtraciju je korišten hsv filter u kombinaciji s roi filterom. Hsv filter se može u sučelju podesiti na (2., 3. i 4.) ili se pritiskom desnog klik na mišu i povlačenjem pokazivača preko željenih stvari za detekciju u (10.) prikazu slike automatski postavljaju vrijednosti za hsv filter. Roi filter se postavlja na (5. i 6.), postavljaju se maksimalne i minimalne vrijednosti pixela koji ulaze u filtraciju te je to vizualizirano tamnijim kvadratom izvan filtera. Postavljanje detekcije je moguće pomoću dijela (7., 8. i 9.) gdje se postavlja maksimalna i minimalna veličina značajke detekcije te je to vizualizirano s dva crna kruga (11.).

# Sustav za praćenje

Namijene sustava za praćenje je određivanje pozicije i orijentacije nekog robota, za ovaj rad specifično površinskih vozila na LABUST-ovom sustavu kamera, no sustav je također napravljen da se može koristiti i na terenu s proizvoljnim sustavom kamera. Te je napravljen da bude lagan i brz za postavit.

## Izbor načina za praćenje

Sustav za praćenje koji je bio izabran je ArUco, to je sistem koji koristi kvadratne markere za određivanje pozicije i orijentaciji, jako je sličan drugim takvim sustavima kao AprilTag, ARTag i drugi, no ArUco je izabran radi najmanje minimalne veličine što daje najveće ‘pixele’ te time i najveću udaljenost na kojoj radi da najmanju površinu samog markera, specifično je izabrana veličina **4x4\_50** što znači da je veličina mreže markera 4x4 te da ih je 50.

## Princip rada

## Implementacija

# Testiranje sustava

# Naslov prvog poglavlja

Rad treba oblikovati kao dokument formata A4 pisan fontom veličine 12 točaka uz prored 1,5 redak, osim na mjestima na kojima je prikladniji drugačiji font i prored (npr., tablice, sheme, itd.).

U ovom poglavlju počnite s pisanjem rada.

## Prvo potpoglavlje

Ovdje se piše tekst.

## Stilovi za tekst, naslove i podnaslove

Stil (engl. style) je rezultat primjene niza naredbi o formatiranju na tekst. Primjenom stilova postižemo ujednačen izgled dokumenta. Za pisanje teksta i naslova u ovom su dokumentu na raspolaganju sljedeći stilovi:

* Normal
* služi za pisanje teksta
* Heading 1
* služi za pisanje naslova poglavlja (npr. 1. Naslov prvog poglavlja)
* Heading 2
* služi za pisanje podnaslova prve dubine (npr. 1.1. Prvo potpoglavlje)
* Heading 3
* služi za pisanje podnaslova druge dubine (npr. 1.1.1. Potpoglavlje niže razine)
* Podnaslov
* služi za pisanje podnaslova treće dubine (nenumerirano)

Napomena: nemojte kreirati dubinu poglavlja ispod treće (npr. 1.1.1.1) jer to nema smisla s obzirom na opseg rada kao takvoga.

## Stilovi za nabrajanje

Prilikom nabrajanja poželjno je koristiti sljedeće stilove.

### Stilovi za nabrajanje s točkama i crticama

Za nabrajanje po jednoj dubini i s točkom na početku koristi se stil:

bullet1

* Stavka 1
* Stavka 2

Ako autor želi dvije dubine nabrajanja, onda se to izvodi s pomoću dva stila:

bullet1 i bullet2

* Stavka 1
* Stavka 11
* Stavka 2

Postoje definirani i stilovi za nabrajanje s brojevima ili slovima:

bullet1brojevi

1. Stavka 1
2. Stavka 2

bullet1slova

1. Stavka 1
2. Stavka 2

Kod pobrajanja po brojevima ili slovima u drugoj dubini, mogu se koristiti sljedeći stilovi:

bullet1 i bulett2brojevi

* Stavka1

1. Stavka 11
2. Stavka 12

* Stavka 2

bullet1 i bullet2slova

* Stavka1

1. Stavka 11
2. Stavka 12

* Stavka 2

## Slike

Prilikom izrade vlastitih crteža i slika (u odgovarajućem alatu ili u samom MS Wordu), treba voditi računa o tome da njihov sadržaj bude jasan i prilikom ispisa na crno-bijelom pisaču (ili fotokopiji). To se najlakše postiže bojanjem u sivim tonovima. Ako koristite razne boje, provjerite čitljivost i u crno-bijelom ispisu. U grafovima umjesto boja ili uz njih možete koristiti različite stilove crta (puna, isprekidana, ili s ugrađenim oznakama).

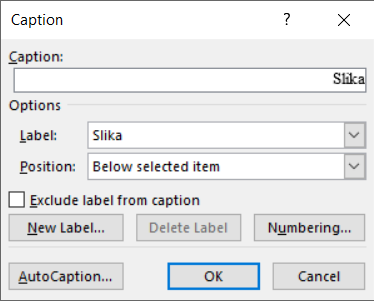
Nakon što ubacite sliku u tekst (kopiranjem ili putem izbornika **Insert/Picture**), selektirajte ju i na nju primijenite stil *slika*. Prema potrebi možete postaviti i položaj slike u odnosu na tekst tako da ponovno odaberete sliku (jednostrukim klikom lijevog gumba miša) i zatim u izborniku (desni gumb miša) odaberite opciju **Format Object**. U opciji **Layout** odaberite **In line with text**.



Sl. . Povezivanje LAN komutatora i javne mreže

Prilikom pripreme slika, preporučuje se da ih odmah pripremite u odgovarajućoj veličini, jer će povećavanje ili smanjivanje razgoditi veličinu slova. Za slike koje crtate u alatu Visio koristite osnovnu veličinu slova Arial 10 pt, ili, ako Vam trebaju manja slova (ovisno o veličini i složenosti slike), koristite Arial 8 pt.

Za ubacivanje kratkog opisa (ispod) slike treba selektirati sliku i u izborniku (desna tipka miša) odabrati opciju **Insert** **Caption**.

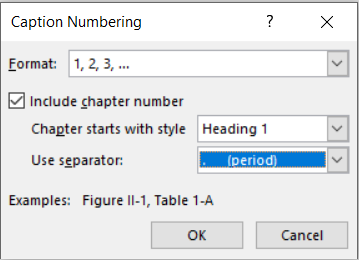


Sl. . Način postavljanja *Captiona* za sliku 1.2

Odaberite oznaku slike (*Label*) Sl. i potvrdite s OK.

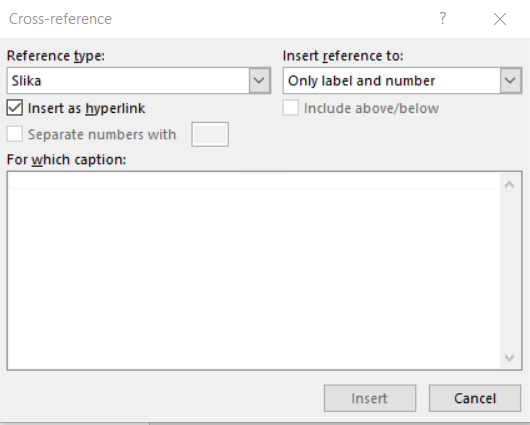
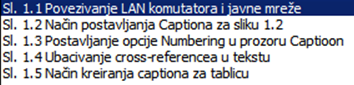
Opcija Insert Caption će ubaciti dio teksta *Sl. <broj>*, nakon čega treba samostalno upisati tekst u nastavku (npr., za sliku 1.1, „Povezivanje LAN komutatora i javne mreže“).

**Napomena:** Kako bi slike bile numerirane kao *<poglavlje>.<broj slike u tom poglavlju>*, opciju **Numbering** u izborniku prozoru **Caption** treba podesiti kao na slici (Sl. 1.3).



Sl. . Postavljanje opcije Numbering u prozoru Captioon

Svaka slika mora biti *referencirana* u tekstu. To znači da se na odgovarajućem mjestu u tekstu „pozivamo“ na sliku, kako bismo na nju usmjerili čitateljevu pažnju. Primjerice, kako bismo se pozvali na sliku 1.1, treba napisati željeni tekst, npr. „Primjer povezivanja čvora lokalne mreže s čvorom javne mreže prikazan je slikom ...“, zatim odaberemo opciju **References** i**Cross-reference**, nakon čega se otvori prozor kao na slici (Sl. 1.4). Odaberite opcije koje su prikazane na slici i kliknite na gumb *Insert*. Rečenica kojom se pozivamo na sliku treba doći **ispred** mjesta u tekstu gdje se slika pojavljuje te tako služi i kao svojevrsna najava (na primjer: „Provedeno je 100 mjerenja, a rezultati su obrađeni i prikazani grafički (Sl. x.y).“)

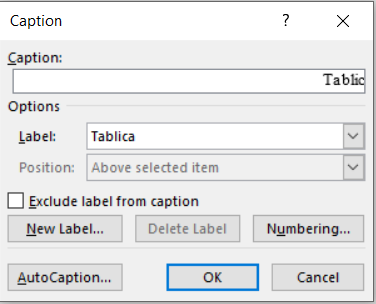


Sl. . Ubacivanje *cross-referencea* u tekstu

Sada će željeni tekst izgledati kao „Primjer povezivanja čvora lokalne mreže s čvorom javne mreže prikazan je slikom (Sl. 1.1).“

## Tablice

Sličan način rada je i s tablicama. Na početku novog odlomka u koji želite postaviti tablicu, ponovo postavite stil *slika*. Umetnite tablicu i popunite stupce u njoj. Dodajte opis (iznad) tablice tako da označite tablicu i odaberete opciju *Insert Caption* (kao *label* odaberite Tablica).



Sl. . Način kreiranja *captiona* za tablicu

U nastavku je prikazan primjer tablice.

Tablica . Parametri prijenosnog linka

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| prijenosna brzina | širina prijenosnog pojasa |  |  |  |
|  |  |  |  |  |

Način referenciranja tablica u tekstu isti je kao i za slike.

## Matematički izrazi i formule

Matematički izrazi obilježava se brojem u zagradama, uz desni rub stranice, a u tekstu se poziva na broj izraza (na primjer: „Kapacitet kanala *C* računa se prema izrazu (1).“)

|  |  |
| --- | --- |
|  | (1) |

Osnovni stil pisanja je Times New Roman 12 pt.

## Programski kôd

Programski kôd može biti označen brojem kao tablica ili slika (ako je opsežniji), ili može biti samo umetnut u tekst (ako se radi o nekoliko redaka).

Više redaka koda (stil *Kôd*):

char \*trazizad(char \*text, char znak) {

  char \*t;

  if(\*text == '\0')

return NULL;

  t = trazizad(text+1,znak);

  if(t != NULL)

return t;

  if(\*text == znak)

return text;

  return NULL;   
}

Kôd . – Program za pronalazak pozicije zadnjeg pojavljivanja znaka u nizu

Nekoliko linija koda:

int nazivVarijable;

nazivVarijable = funkcijaRacunajFaktorijele(5);

Ako se neki kod (npr. naziv klase, varijable ili metode), npr. nazivVarijable pojavljuje unutar teksta onda se to treba označiti stilom *Kôd u tekstu*.

# Zaključak

Na kraju rada piše se kratak zaključak, duljine do najviše jedne stranice.

# Literatura

Popis literature dolazi na kraju rada, iza zaključka, a prije ostalih priloga.

Na naslov **Literatura** primijenite stil Heading 1, a zatim ručno maknite brojčanu oznaku (to je važno kako bi i naslov „Literatura“ ušao u sadržaj na početku rada, prije uvoda).

Pri kreiranju navoda u popisu literature koristite stil *literatura*.

Primjeri u nastavku ilustriraju navođenje raznih izvora u popisu literature: (1) knjige, (2) članka u časopisu, (3) članka u zborniku konferencije, (4) doktorskog, magistarskog ili diplomskog rada, (5) web-stranice.

1. Tanenbaum, A. S., Wetherall, D. J. *Computer Networks*. 5. izdanje. London: Pearson, 2013.
2. Brady, P.T. *A Statistical Analysis of On-off Patterns in 16 Conversations*, Bell System Technical Journal, 47,1 (1998), str. 55-62.
3. Brady, N. *A Statistical Analysis of Use Case*. Proceedings of the 7th International Conference on Telecommunications ConTEL, Zagreb, (2003), str. 45-52.
4. Ivić, M. *Analiza ponašanja korisnika u digitalnim igrama namijenjenima učenju*. Diplomski rad. Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva, 2016.
5. Epstein M., The *best VR headset in 2019*, PC Gamer, (2019, listopad). Poveznica: <https://www.pcgamer.com/best-vr-headset/>; pristupljeno 4. listopada 2019.

Uz svaki preuzeti sadržaj u svom radu – bilo da je riječ o tekstu (izravno citiranome ili „prepričanome“), slici ili grafičkom prikazu – treba navesti oznaku izvora (članak, knjiga, web-stranica ...) u popisu literature te se na nju „pozvati“, na primjer:

Međusobno povezivanje mreža zasniva se na primjeni komunikacijskih protokola (Tanenbaum i Wetheral, 2014).

Podaci o karakteristikama uređaja za virtualnu stvarnost preuzeti su s portala PC Gamer [5].

Početna verzija programa preuzeta je iz diplomskog rada [5].

U danim primjerima mogli ste uočiti dva načina referenciranja:

* (Tanenbaum i Wetheral, 2014),
* [1].

Kad izaberete jedan od njih svakako ga se držite konzistentno u cijelome radu.

# Sažetak

Naslov, sažetak, ključne riječi (na hrvatskom jeziku)

Sažetak opisuje sadržaj rada, prepričan u stotinjak riječi.

# Summary

Title, summary, keywords (na engleskom jeziku)

# Skraćenice

Ovo poglavlje nije obavezno, ali se može dodati radi preglednosti.

ATM *Asynchronous Transfer Mode* asinkroni način prijenosa

ISDN *Integrated Services Digital Network* digitalna mreža integriranih usluga

Napomena: na naslov **Skraćenice** primijenite stil Heading 1, a zatim ručno maknite brojčanu oznaku (to je važno kako bi i skraćenice ušle u sadržaj na početku rada, prije uvoda). Pri kreiranju popisa skraćenica koristite stil *nabrajanje*.

# Privitak

Privitak je također opcionalno poglavlje (u dogovoru s mentorom).

Sadržaj koji se stavlja u privitak je, općenito, nešto što je, kao cjelinu, prikladno izdvojiti iz sadržaja samog rada.

Mogući primjer je tehnička dokumentacija vezana uz završni rad - npr. električka i položajna shema sklopa, sastavnica, predložak tiskane veze, plan bušenja, ispis programa s detaljnim opisom.

Drugi primjer uključuju upute za korištenje rezultata rada (softvera ili hardvera), detaljni ispisi mjerenja čiji su rezultati sažeto ili grafički prikazani u radu. Ako se radi o softveru, uobičajeno je navesti podatke o platformi na kojoj se izvodi (npr., karakteristike uređaja i operacijskog sustava te pomoćnog softvera), kao i upute za instalaciju.

U privitku nemojte koristiti stilove razine Heading, već samo (nenumerirani) stil Podnaslov.

Na primjer:

Instalacija programske podrške

Upute za korištenje programske podrške

.

.

.

Ostali savjeti

U izborniku Tools - opcija Language postavite Croatian kao *default* jezik. Na kraju provedite strojnu provjeru teksta (*spell checking*, ako ga imate ugrađenoga), ali svakako pažljivo i pročitajte vlastiti tekst.

Ako nemate ugrađeni *spelling checker* za hrvatski jezik, možete se poslužiti Hascheckom (izgovara se Hašek, a ime dolazi od kratice za Hrvatski akademski *spelling checker*), dostupan putem poveznice <https://ispravi.me/>.