Vizijski sustav za automatsku kontrolu kvalitete uložaka grebenaste sklopke

Rešetar, Emanuela

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

DIPLOMSKI RAD

Emanuela Rešetar

Zagreb, godina 2020.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

DIPLOMSKI RAD

Mentor:

Doc. dr. sc. Tomislav Staroveški

Student:

Emanuela Rešetar

Zagreb, 2020.

Izjavljujem da sam ovaj rad izradila samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Ovim se putem želim zahvaliti svome mentoru, doc. dr. sc. Tomislavu Staroveškom koji je bio stručna, tehnička i moralna podrška pri izradi ovog diplomskog rada, ali i cijelog diplomskog studija. Njegova predanost, stručnost i pristup čine ga osobom i znanstvenikom u koju se svaki student našeg fakulteta jednog dana želi razviti. Zahvaljujem se i svom nastavnom osoblju FSB-a na pruženom znanju stečenom kroz diplomski studij.

Hvala mojim roditeljima, ocu Borisu i majci Marini, kojima posvećujem ovaj diplomski rad. Njihova žrtva je velikim dijelom omogućila moje akademsko obrazovanje, te je ovaj diplomski rad mali dio kojim želim zahvaliti za svu pruženu podršku i ljubav, svih ovih godina. Hvala mojem bratu Josipu koji me naučio kako vjerovati, uvijek ustrajati i nikad ne odustati od svojih snova.

Također hvala mojim prijateljicama, Silviji, Senki i Andrijani koje su bile uz mene kada je bilo potrebno a bez kojih budućnost izgleda nezamislivo.

Hvala mojem Marcu, koji je bio velika podrška te učinio moj diplomski studij ljepšim i uspješnijim.

Emanuela Rešetar

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za diplomske radove studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment,
inženjerstvo materijala te mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum:	Prilog:
Klasa: 602 - 04 / 20 - 6 / 3	
Ur. broj: 15 - 1703 - 20 -	

DIPLOMSKI ZADATAK

Student: **EMANUELA REŠETAR** Mat. br.: 0035193795

Naslov rada na hrvatskom jeziku: **Vizijski sustav za automatsku kontrolu kvalitete uložaka grebenaste sklopke**

Naslov rada na engleskom jeziku: **Machine vision system for automated quality control of rotary switch inserts**

Opis zadatka:

Na liniji za proizvodnju grebenastih sklopki postoji nedostatak povezan s kontrolom kvalitete uložaka grebenastih sklopki. Oblici predmetnih uložaka definiraju tip grebenaste sklopke i mogu se smatrati ključnim elementima tog proizvoda. Predmetni elementi proizvode se ciklički injekcijskim prešanjem i zbog svoje međusobne geometrijske sličnosti u procesu sklapanja često dolazi do njihove zamjene, što u konačnici rezultira neispravnom sklopkom. U cilju cjelovite automatizacije ovog procesa, nužno je osmisliti odgovarajući vizijski sustav koji bi mogao raspoznavati predmetne elemente u nesređenoj okolini i time ukloniti potrebu za ručnim razvrstavanjem.

U radu je potrebno:

1. Dati pregled potencijalno prikladnih algoritama prepoznavanje i lokalizaciju objekata na nesređenim dvodimenzionalnim scenama.
2. Odabrati i testirati elemente vizijskog sustava s obzirom na oblike i moguće orijentacije ispitnih uzoraka.
3. Primjenom biblioteka otvorenog koda (engl. Open Source), predložiti i napisati algoritam za robusno prepoznavanje i kontrolu ispitnih uzoraka.
4. Testirati sustav na više scena s različitim kombinacijama razmještaja ispitnih uzoraka.
5. Dati zaključke rada.

U radu je potrebno navesti korištenu literaturu i eventualno dobivenu pomoć.

Zadatak zadan: 24. rujna 2020.

Rok predaje rada: 26. studenog 2020.

Predviđeni datum obrane: 30. studenog do 4. prosinca 2020.

Zadatak zadao:

doc. dr. sc. Tomislav Staroveški

Predsjednica Povjerenstva:

prof. dr. sc. Biserka Runje

SADRŽAJ

POPIS SLIKA

POPIS TABLICA

POPIS OZNAKA

Kratica	**Izvorni naziv**
AIA	*Global Association for Vision Information* – Globalno udruženje za informacije o viziji
CV	*Computer Vision* – računalni vid
CCD	*Charge Couple Device* – uređaj spojen na punjenje
CMOS	*Complementary Metal Oxide Semiconductor* – komplementarni poluvodič od metalnog oksida
ADU	*Analog Digital Unit* – analogno-digitalna jedinica
ADC	*Analog to Digital Conversion* – analogno-digitalna pretvorba
DSLR	*Digital Single Lens Reflex Camera* – digitalni jednooki zrcalno-refleksivni fotoaparat
RAW	*Read and write* – čitaj i piši
RGB	*Red Green Blue* – crveno zeleno plavo
LPF	*Low Pass Filter* – niskopropusni filtar
HPF	*High Pass Filter* – visokopropusni filtar
PDE	*Partial Derivative Equation* – parcijalna diferencijalna jednadžba
ANN	*Artificial Neural Network* – umjetna neuronska mreža
NMS	*Non Maximum Suppression* – potiskivanje ne maksimuma
LoG	*Laplacian of Gaussian*
MSE	*Mean Squared Error* – srednja kvadratna pogreška
PSNR	*Peak Signal to Noise Ratio* – vršni omjer signala i šuma
NBD	*Negative Binomial Distribution* – negativna binomna distribucija
OpenCV	*Open Source Computer Vision Libary* – biblioteka računalnog vida otvorenog koda
MATLAB	*Matrix Laboratory* – matrični laboratorij
BSD	*Berkeley Software Distribution* – Berkeley softverska distribucija
OSX	*Operating System X* – operativni sustav X
2D	*dvodimenzionalan*
3D	*trodimenzionalan*
macOS	*Macintosh Operating System* – Machintosh operativni sustav

BMP	*bitmap image* – rasterska slika
VDC	*volts direct current* – volti istosmjernog napona
I/O	*input/output* – ulaz/izlaz
H	*height* – visina
W	*width* – širina
L	*length* – dužina
IDE	*Integrated Development Environment* – Integrirano razvojno okruženje

SAŽETAK

Rad se bavi razvojem vizijskog sustava za prepoznavanje i lokalizaciju uložaka grebenaste sklopke. Predstavljene su dvije algoritamske metode prikladne za robusno prepoznavanje. Prva algoritamska metoda koristi Houghovu kružnu transformaciju na binarnim slikama kako bi se locirali uzorci, koji se zatim kategoriziraju analizom intenziteta obodnih piksela. Druga algoritamska metoda temelji se na segmentaciji slike i analizi bloba pri čemu se koriste momenti za lokalizaciju i analiza konture za prepoznavanje. Oba pristupa testirana su na velikom broju slika različite složenosti (u rasponu od jednostavnih slika bez preklapanja uzoraka do slika sa znatnim preklapanjem uzoraka). Rezultati pokazuju kako su obje algoritamske metode prikladne za robusno prepoznavanje i lokalizaciju uzoraka. Međutim, prvi algoritam pokazao se bolji u pogledu vremena odziva i računske intenzivnosti.

Ključne riječi: strojni vid, računalni vid, procesiranje slike, Houghova transformacija, detekcija kontura, OpenCV

SUMMARY

This thesis presents the development of a machine vision system for the classification and localization of rotary switch inserts. In order to achieve robust results, two approaches were investigated. In the first approach, Hough transform for circles was used on binarized edge images in order to localize inserts, which were then classified by analyzing pixel intensities across the insert perimeter. The second approach is based on the image segmentation and blob analysis in which moments were used for localization and contour analysis for classification. Both approaches were tested on a large number of images with different scene complexities (ranging from simple images with no insert overlapping to images with considerable insert overlapping). Results show that both approaches can be used for this application, however, the first approach had a lower response time and was less computationally intensive.

Key words: Machine vision, Computer Vision, Image processing, Hough Transformation, Detection of Contours, OpenCV

1. UVOD

Rad se bavi razvojem vizijskog sustava za automatsku kontrolu uložaka grebenaste sklopke. Kod procesa proizvodnje grebenastih sklopki dolazi do problema zamjene proizvoda. Uzrok ovog problema leži u sličnosti oblika uložaka grebenastih sklopki. Predmetni oblici dobivaju se injekcijskim prešanjem i zbog svoje geometrijske sličnosti dolazi do zamjene istih što u konačnici rezultira montažom neispravnog sklopa. Geometrijske karakteristike ispitivanih uložaka predstavljaju ključni element proizvoda. Kako bi eliminirali mogućnost grešaka i omogućili bolju kontrolu kvalitete, u okviru ovog diplomskog rada razvijen je sustav prepoznavanja ispitivanih elemenata primjenom vizijskih sustava čime se otklanja potreba za ručnim razvrstavanjem. Zbog širine predmetnog područja u okviru diplomskog rada detaljno su obrađeni samo oni algoritmi za obradu i analizu slike koji su potencijalno prikladni za rješavanje zadanog problema.

U prvom dijelu rada, kojeg čine poglavlja 2 i 3, opisana je važnost primjene strojnog vida te osnovni pojmovi. Napravljena je detaljna analiza postupka obrade slike i dani su pregledi glavnih tehnika potrebnih za daljnji razvoj algoritma.

Drugi dio rada, koji obuhvaća 4. poglavlje, sadrži pregled glavnih metoda za izdvajanje značajki tijekom analize slike. Prikazan je pregled metoda od interesa za radni zadatak te su odabrani najprikladniji algoritmi i tehnike za postupak detekcije i identifikacije zadanog predmetnog uzorka.

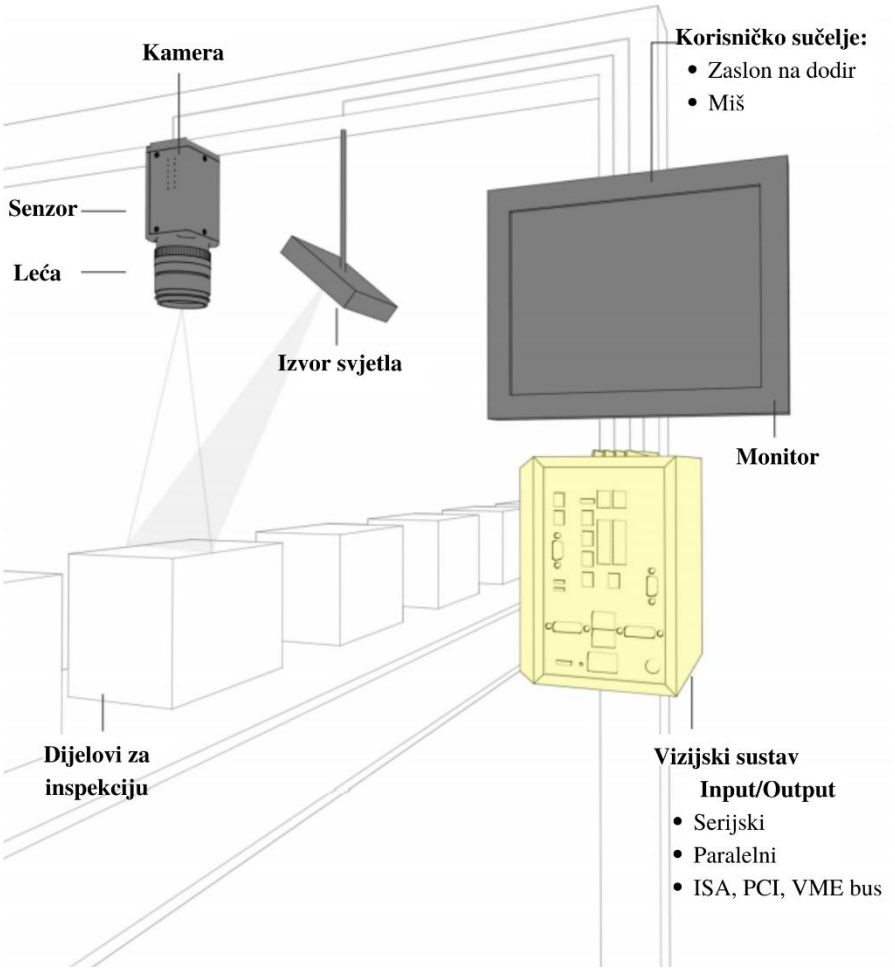
U trećem dijelu, koje obuhvaća poglavlje 5, prikazan je pregled predloženih algoritama za raspoznavanje i lokalizaciju objekata na nesređenim dvodimenzionalnim scenama. Algoritmi su testirani obzirom na oblike, moguće orijentacije ispitivanih uzoraka, njihov međusobni položaj. U konačnici su izneseni zaključci.

2. TIJEK PROCESA U SUSTAVU STROJNOG VIDA

Prema AIA (engl. *Global Association for Vision Information*): "strojni vid obuhvaća sve industrijske i neindustrijske primjene, u kojima se kombinacijom hardvera i softvera, osigurava procesno vođenje uređaja za izvršavanje funkcije koje se zasnivaju na snimanju i procesiranju slike". Cilj strojno vođenih vizijskih sustava je dakako, razumijevanje viđenog, pri čemu se uz pomoć algoritama koji koriste sliku kao ulaznu informaciju, pruža simbolička interpretacija o objektima na slici, određuje njena orijentacija te dobivaju informacije o traženim odnosima u trodimenzionalnom prostoru. Točnije, strojni vid može klasificirati, identificirati, verificirati i detektirati objekte.

Informacije dobivene obradom slike uz primjenu računalnog vida mogu biti jednostavan podatak koji daje izvješće da li se radi o dobrom odnosno lošem dijelu ili pak kompleksan set informacija o predmetu, njegovoj poziciji i orijentaciji neovisno o sređenosti scene. Strojni vid u današnjici predstavlja nezamjenjiv alat koji omogućava izgradnju fleksibilnijih visoko automatiziranih sustava [1].

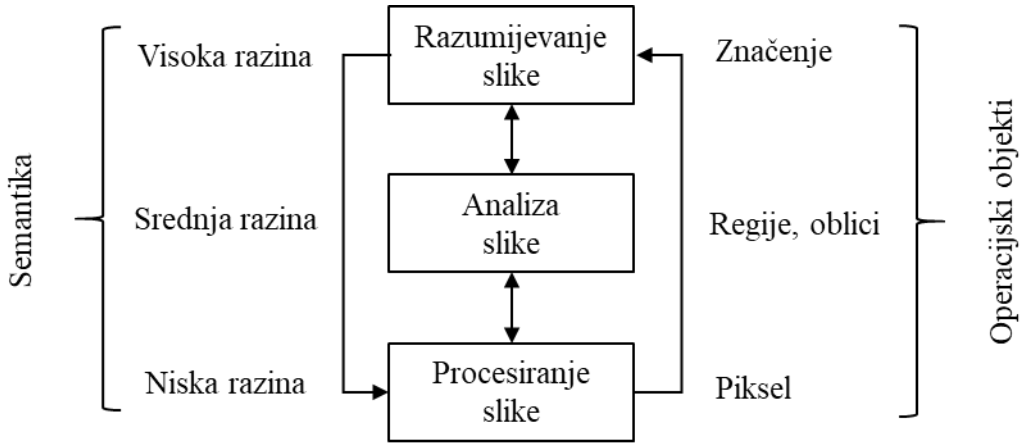
Glavne komponente vizijskog sustava prikazane su na slici 2.1.



Slika 2.1 Glavne komponente sustava za strojnu viziju [1]

2.1. Računalni vid

Računalni vid (engl. *Computer Vision; CV*) predstavlja interdisciplinarno područje, koje računalu pruža sposobnost identifikacije, procesiranja i razumijevanja objekata na sceni. Na slici 2.2 prikazan je tok rada računalnog vida te dijelovi koji će biti obrađeni u radu.



Slika 2.2 Obrada slike putem računalnog vida

2.2. Tijek informacija u sustavu računalnog vida

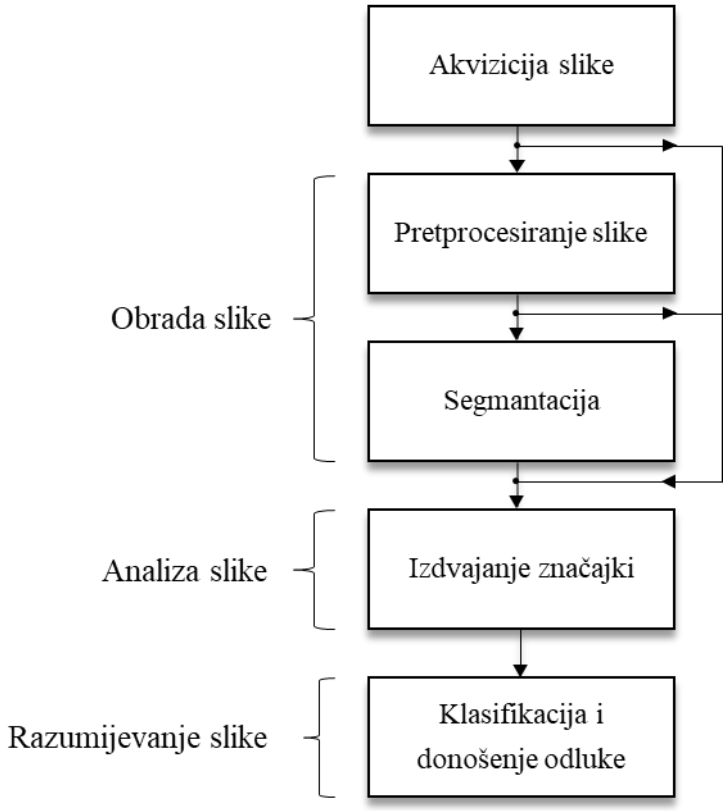
Akvizicija slike je prvi korak u automatskoj inspekciji strojnim vidom za koju se obično koriste kamere, leće i osvjetljenje dizajnirano kako bi osiguralo snimanje slike pogodne za daljnju obradu. Važan preduvjet dobrih izlaznih parametara je upravo ispravno određivanje uvjeta prikupljanja slikovnih podataka.

Digitalni slikovni zapis potom se podvrgava procesu digitalne obrade. Provodi se postupak pretprocesiranja koji uključuje postupke i metode prilagođavanja slike pri kojima ispunjava zadane kriterije. Sadržaj slike smanjuje se na željeni nivo naglašavanjem odnosno potiskivanjem određenih slikovnih podataka. Poboljšavanje se može ostvariti u pogledu kontrasta i rubova, uklanjanja šuma ili izoštravanje.

Postupak segmentacije bavi se dekompozicijom scene u njezine sastavne dijelove. Segmentirane regije imaju iste željene karakteristike. Cilj segmentacije je razdvajanje bitnih od nebitnih značajki čime se znatno reducira količina nepotrebnih podataka te osigurava uspješnija analiza.

Izdvajanje značajki predstavlja početni korak pri analize slike te ima važnu ulogu pri klasifikaciji sadržaja. Ulazne informacije digitalne slike nakon provedenog izdvajanja značajki prikazane su njihovim značajnim podacima. Kod procesa analize te prepoznavanja nije potrebno poznavati sva svojstva objekta kojeg se analizira, nego značajne podatke potrebno za klasifikaciju. Značajka može biti bilo koja karakteristika objekta.

Na osnovu poznatih značajki donosi se odluka stanju promatrane scene. Tijek procesa računalnog vida prikazan je na slici 2.3.

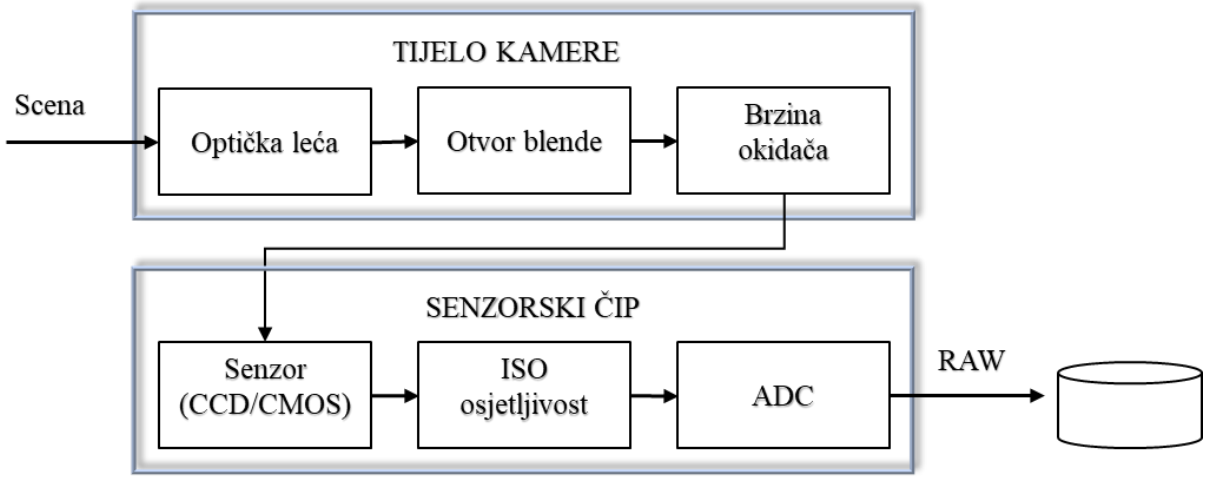


Slika 2.3 Tijek procesa u sustavu računalnog vida [2]

2.3. Akvizicija

Cilj procesa akvizicije slike je transformiranje optičke slike (stvarne fizičke tvorevine) u niz numeričkih podataka kojima je moguće manipulirani na računalu. Slika se snima odgovarajućom kamerom. Senzor kamere prima određen oblik mjerljive energije pomoću koje je moguće ostvariti snimanje promatrane scene. Energija interesa u ovom kontekstu je svjetlo.

Slika je generirana kombinacijom izvora svjetla i vezanom refleksijom ili apsorpcijom energije promatranih elementa na sceni koja se snima. Leća unutar kamere je nositelj informacije o podacima slike koju prenosi do senzora. Brzina okidača i otvor blende direktno kontroliraju količinu svijetlosti, odnosno broj fotona, koja dolazi do senzora koji je u proporcionalnoj vezi s ekspozicijom slike. Prikaz operacija unutar kamere u procesu akvizicije prikazan je na slici 2.4.



Slika 2.4 Slijed operacija unutar kamere

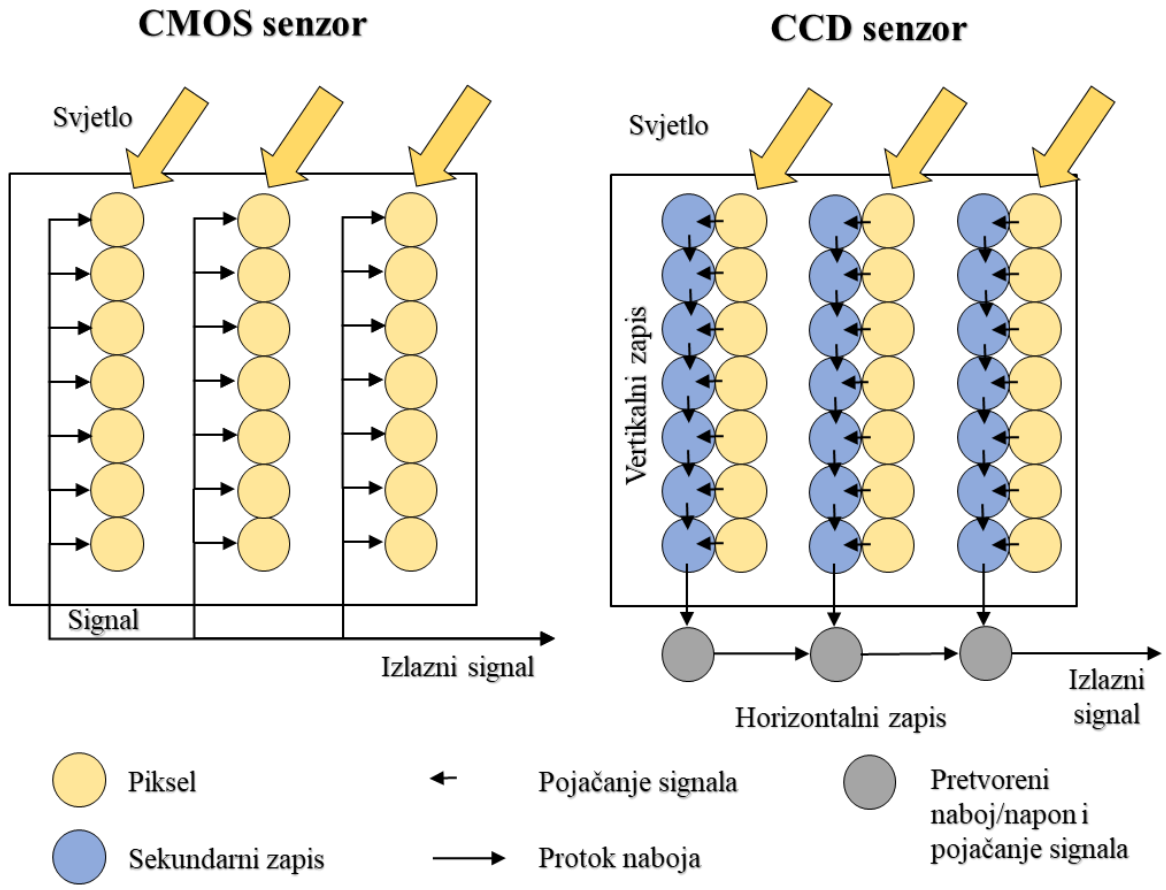
U industrijskoj primjeni koriste se digitalne kamere u kojima nalaze se dva tipa senzora :

1. CCD senzor (engl. *Charge Couple Device*).
2. CMOS senzor (engl. *Complementary Metal Oxide Semiconductor*).

CCD senzori pomiču fotogenerirani naboj od piksela do piksela, nakon čega se on pomoću nabojnih pojačala pretvara u napon, te A/D pretvorbom u numeričku vrijednost piksela. Iako sporiji, CCD senzori hvataju cijelu sliku, takozvani *"global shutter"*, odjednom čime se izbjegava zamućenje slike uzrokovano pokretom (engl. *motion blur*). Nedostatak koji se javlja kod CCD senzora je cvjetanje (engl. *blooming*), efekt koji nastaje protjecanjem naboja jednog piksela na susjedne piksele, nakon čemu je podatak sa danog piksela prisutan u susjednim pikselima.

CMOS senzori koriste jedno pojačalo po fotodiodi. Na taj se način vrijednosti piksela očitavaju pojedinačno što značajno ubrzava proces očitavanja te se izbjegava učinak cvjetanja. Međutim, brzim očitavanjem svjetlosnih informacija uzrokuje se takozvani *"rolling shutter efekt"*. Ovaj efekt je uzrokovan jer se različiti dijelovi senzora izlažu svijetlu u različito vrijeme. Nedostatak izloženog efekta je zamućenje slike kod ekstremno brzih pokreta ili bljeskova.

CMOS senzor je češće korišten kod digitalnih kamera, zbog prihvatljivije cijene te jednostavnijeg proizvodnog procesa. Rad senzora prikazan je na slici 2.5.



Slika 2.5 Prikaz rada CMOS i CCD senzora [3]

Senzori unutar fotodiode korištenjem fotoelektričnog efekta detektiraju fotone koje djeluju na energijsku promjenu stanja energije elektrona. Po završetku ekspozicije senzora akumulirani naboj se očitava i transformira u digitalnu sliku prema izrazu (2.1)

$$Broj\ detektiranih\ fotona = Broj\ izmjerenih\ elektrona = Gain \cdot ADU \tag{2.1}$$

Pri čemu je:

ADU – (engl. *Analog-Digital Unit*) Očitane vrijednosti piksela.

Gain – Pojačanje – parametar evidentiran u tehničkim specifikacijama kamere[5].

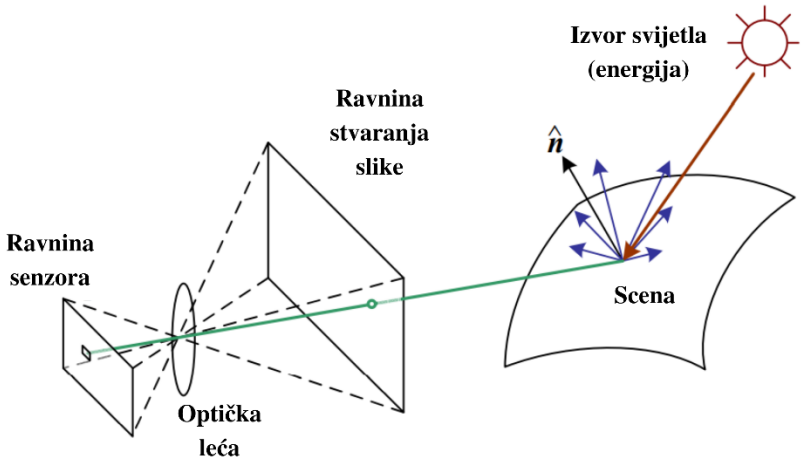
Sama ideja je da se ulazna energija svijetlosti transformira u naponski signal kombinacijom ulazne električne energije i senzora materijala koji daje odziv na specifičnu ulaznu energiju. Izlazni signal predstavlja odziv senzora koji je digitalan.

Prije analogno-digitalne pretvorbe, snimljeni signal je obrađen putem ISO osjetljivosti ili fotoosjetljivosti matrice, odnosno filma. Kao takva označava parametar koji determinira razinu osjetljivosti elementa prema akumuliranju svjetlosti.

Završni korak u lancu analogno-digitalnog procesiranja je analogno digitalna-pretvorba ADC (engl. *Analog to Digital Conversion*). U okviru spomenute pretvorbe dva su parametra od značaja pri procesu :

- Rezolucija – Količina bita u slici.
- Razina šuma – Količina korisnih bita za proces obrade.

Kod većine kamera broj bita označava raspon intenziteta piksela koje je na slici moguće detektirati. Prvi korak za kvalitetnu akviziciju je jednostavan postupak kalibracije šuma repetitivnim snimanjem željene scene te predočavanjem šuma kao funkcije svijetla. Na slici 2.6 prikazan je pojednostavljen model postupka akvizicije slike. Važno je spomenuti kako pojednostavljeni model ignorira višestruke refleksije koje su česta pojava u stvarnim scenama



Slika 2.6 Pojednostavljeni model fotometrijske akvizicije slike. [5]

3. DIGITALNA OBRADA SLIKE

Digitalna obrada slike (engl. *Digital Image Processing*) obuhvaća tehnike podvrgavanja slike numeričkim operacijama s ciljem postizanja željene kvalitete po nekom određenom kriteriju. U postupku digitalne obrade slikovni podatak nalazi se na ulazu i izlazu (Slika 3.1) [4].

Slika 3.1 Ulazna i izlazna informacija je istog formata – slikovni podatak [4]

Nakon postupka akvizicije slike kroz interakciju promatrane scene, osvjetljenja, optičke kamere i senzora, dolazi prva faza većine algoritama prikladnih za primjenu računalnog vida. Primjeri operacija koje se u ovom koraku često upotrebljavaju uključuju korekciju i balansiranje boje izjednačavanjem histograma, redukciju šuma, povećanje oštrine ili transformacije slike. Digitalna slika, odnosno digitalizirani slikovni podatak, nalazi su na ulazu i izlazu procesa.
Pojam digitalna slika predstavlja uzorkovanje i kvantizaciju dvodimenzionalne realne slike. Izlazne informacije digitalne slike mogu biti prikazane kao niz funkcija *f*(*m,n*), gdje su *m i n* planarne koordinate, dok *f* predstavlja intenzitet slike u točki na položaju (*m,n*). Postupak uzorkovanja slike je digitaliziranje vrijednosti koordinata, dok kvantizacija predstavlja digitaliziranje vrijednosti amplituda. Slika postaje digitalna kada *m, n i f* postanu konačne diskretne vrijednosti [6].

Digitalna fotografija može biti prikazana kao dvodimenzionalna matrica prema izrazu (3.1):

$$F = \left[\begin{pmatrix} f(1,1) & \cdots & f(1,n-1) \\ \vdots & \ddots & \vdots \\ f(m-1,1) & \cdots & f(m-1,n-1) \end{pmatrix}\right] \tag{3.1}$$

Pri čemu je:

F – matrični zapis digitalne slike s m redaka i n stupaca

$f\,(m, n)$ – slikovni element, piksel .

Vrijednost f proporcionalna je svjetlini (ili razini zatamnjenja) slike u određenoj točki. Vrijednost intenziteta kreće se u rasponu prikazanom u (3.2).

$$0 < f(x, y) < \infty \tag{3.2}$$

Funkcija intenziteta osvjetljenja pod izravnim je utjecajem apsorbirane ili reflektirane svjetlosti. Svaki element slike ima konačnu, diskretnu numeričku vrijednost intenziteta svjetline određene boje pri specifičnoj koordinati. Ovi elementi se nazivaju pikseli. Boja se predstavlja trodimenzionalnom matricom dimenzija $M \times N$ pomnoženom s brojem bita b potrebnih za pohranu razine intenziteta ($L = 2b$). Veličina slike izražava se prema izrazu (3.3):

$$M \times N \times b \tag{3.3}$$

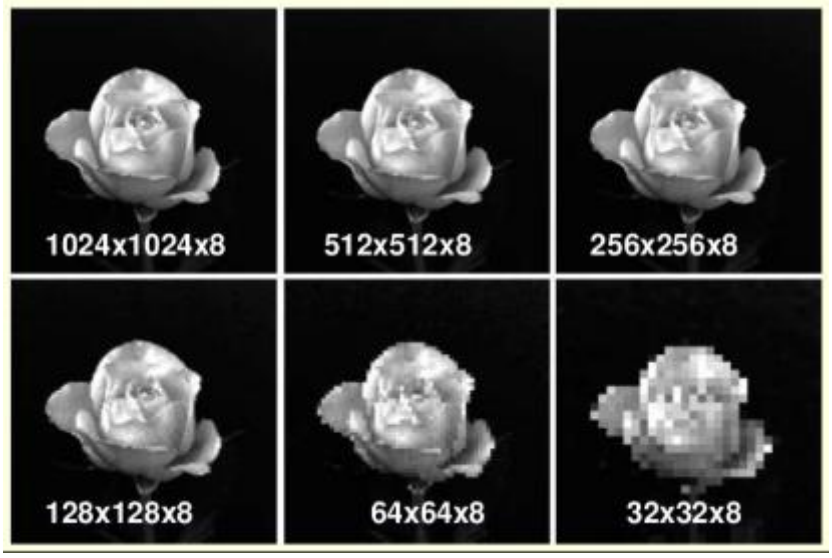
Uobičajene vrijednosti b su:

b = 1 - binarne slike (crno i bijelo)

b = 8 - za monokromatske (255 tonova sive boje) ili indeksirane slike u boji

b = 24 - za RGB slike.

Digitalna slika posjeduje konačan broj digitalnih vrijednosti dodijeljenih elementima slike, pikselima, čime formira rasterski format slike. Pikseli su pohranjeni u memoriji računala kao rasterska slika ili rasterska mapa, dvodimenzionalna matrica cijelih brojeva. Ove vrijednosti su obično prenesene ili pohranjene u kompresijskom obliku. Rezolucija slike je određena brojem piksela na slici 3.2 [7].



Slika 3.2 Usporedba kvalitete slike u ovisnosti o smanjenju rezolucije [7]

U ovom radu bit će obrađena obrada i analiza za monokromatske (engl. *gray – scale*) slike.

Monokromatske slike su iskazane prema intenzitetu sive između 0-255 kod monokromatske ili 0-1 kod binarne slike. Potrebno je 8 bita kako bi pohraniti monokromatsku vrijednost. Svaki piksel poprima vrijednost 0 (crna) i 255 (bijela) što u konačnosti determinira intenzitet sive boje.

3.1. Pretprocesiranje

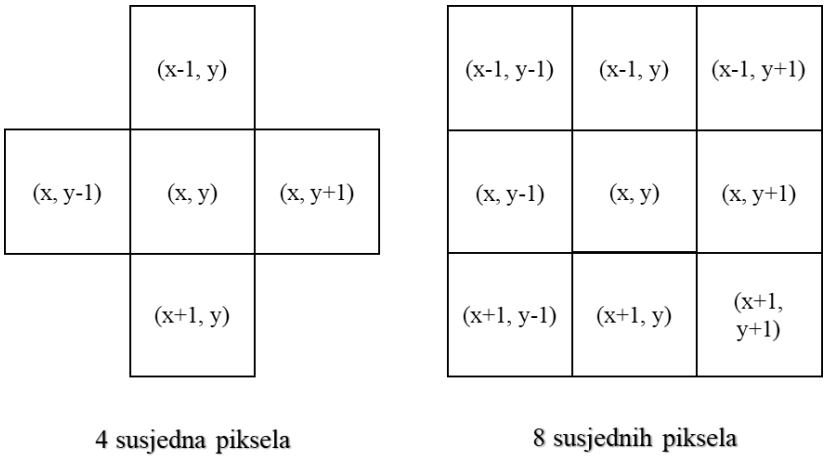
Pretprocesiranje slike je postupak pripreme slike koja će biti pogodna za daljnju obradu i analizu. Operacije koje se provode sužavaju sadržaj informacija na željeni nivo interesa. Glavi cilj pretprocesiranja je poboljšanje slike koje radi na način da potiskuje neželjena izobličenja ili naglašava tražene značajke slike relevantne za daljnju obradu i analizu. Time se u prvom redu misli na filtriranje šuma i poboljšanje kontrasta u svrhu olakšavanja primjene kasnijih operacija. Proces filtriranja digitalne slike moguće je izvesti u frekvencijskoj i prostornoj domeni. Dok prostorni algoritmi vrše direktnu manipulaciju nad pikselima u slici, frekvencijski se baziraju na Fourierovoj transformaciji slike [6, 8].

3.2. Filtriranje u prostornoj domeni

Filtriranje u prostornoj domeni je tehnika prostornog ujednačavanja putem naglašavanja ili uklanjanja određene značajke slike. Postupak procesiranja slike uključuje operacije kao što su zaglađivanje, izoštravanje i određivanje rubova. Osnovna ideja je da se jedna slika konačnih dimenzija i oblika konvulira preko ulazne digitalne slike nakon čega se izlazne vrijednosti piksela izračunavaju prostornim ujednačavanjem, odnosno računanjem konvolucije između slike i maske. Slika kojom se vrši konvolucija posjeduje određen oblik s pridruženim vrijednostima piksela (težinama) i naziva se konvolucijska maska. Konvolucijska maska pripada linearnom prostornom filtru i ima veličinu u pravilu manju od veličine slike pri čemu je broj redova i stupaca neparan broj (primjerice 3×3, 5×5, 7×7).

3.2.1. Linearno prostorno filtriranje

Linearno prostorno filtriranje je vrsta filtriranja slike u prostornoj domeni koja se ostvaruje kroz već spomenutu konvoluciju. Filtriranje se provodi direktno na pikselima slike pri čemu se svaki element lociran ispod maske množi s koeficijentima maske i zatim iskazuje kao težinska suma susjednih piksela. Susjedni pikseli prikazani se na slici 3.3.



Slika 3.3 Prikaz položaja i vrijednosti susjednih piksela

Udaljenost između susjednih piksela

Udaljenost između dva piksela definirana je Euklidovom udaljenosti prema izrazu (3.4):

$$D_{\mathrm{E}}(p_1, p_2) = \sqrt{[(x_2 - x_1)^2 + (y_2 - y_1)^2]} \tag{3.4}$$

Pri čemu je vrijednost prvog piksela iskazana kao $p_1 = (x_1, y_1)$ a vrijednost drugog piksela $p_2 = (x_2, y_2)$.

Prikazana formula može dati rezultat koji neće biti cijeli broj, jer promatra vrijednost najkraće udaljenosti dvaju točaka koja se ne može poistovjetiti sa mjerenjem udaljenosti susjednih piksela. Obzirom na konkretnu problematiku, izvršava se aproksimacija euklidske udaljenosti prema izrazima (3.5) i (3.6):

$$D_{Manh}(p_1, p_2) = |x_2 - x_1| + |y_2 - y_1| \tag{3.5}$$

Prikazani izraz poznatiji je pod nazivom udaljenost gradskog bloga (engl. *City Block Distance*) ili Manhattan udaljenost koja daje cjelobrojnu vrijednost udaljenosti. Sljedeća aproksimacija primjenjuje se kod problema definiranja udaljenosti kvadratnom matricom te je poznata kao udaljenost šahovske ploče (engl. *Chessboard Distance*):

$$D_{\mathrm{Chess}}(p_1, p_2) = \max(|x_2 - x_1|, |y_2 - y_1|) \tag{3.6}$$

Za koji vrijede sljedeće vrijednosti prema izrazima (3.7) i (3.8):

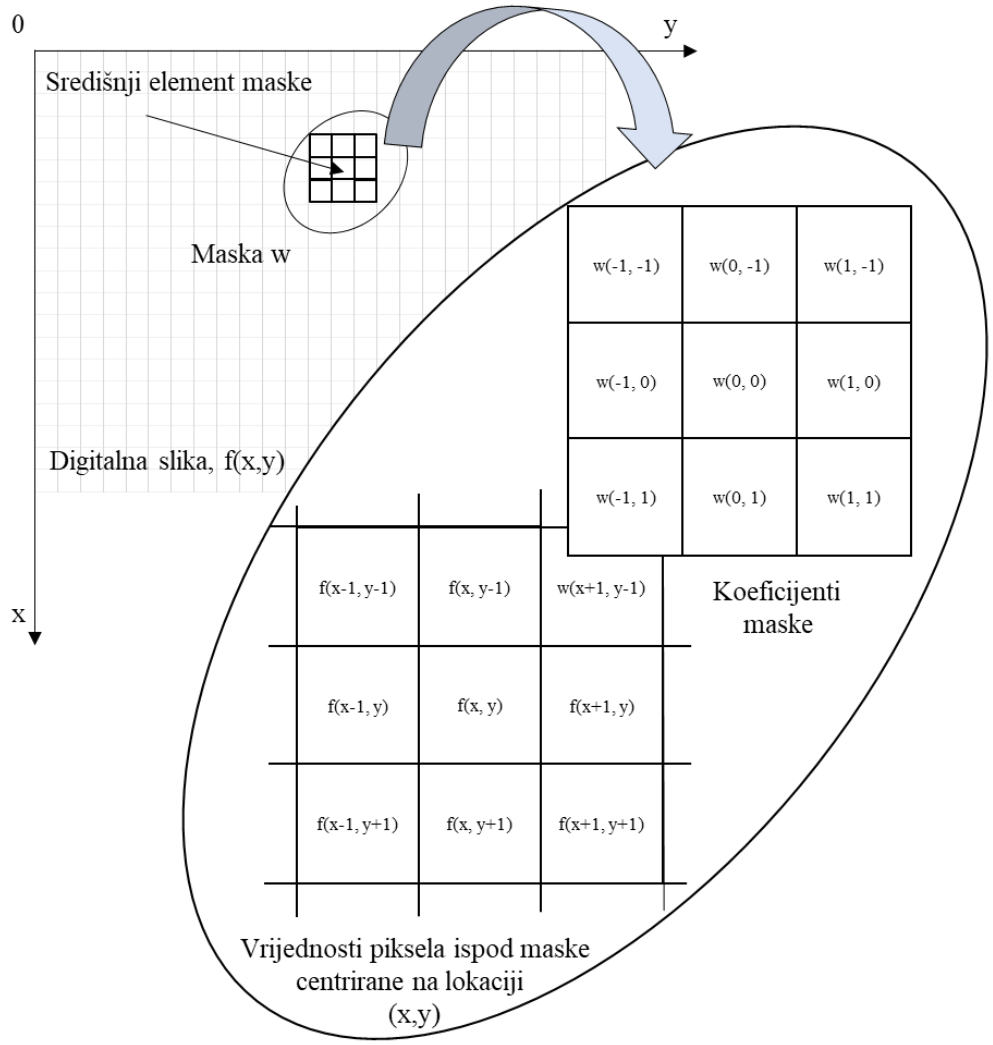
$$\begin{matrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{matrix} \tag{3.7}$$

Gdje je udaljenost piksela $D_{\text{Chess}} = 1$. Udaljenost je iskazana za iznos matrice 3×3 te je središnji element okružen s 8 susjednih piksela. U slučaju primjene matrice veličine 5×5, prikazane u (3.8) dobiva se udaljenost susjednih piksela $D_{\text{Chess}} = 2$.

$$\begin{matrix} 2 & 2 & 2 & 2 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{matrix} \tag{3.8}$$

Mehanizam linearnog filtriranja

Linearna transformacija slike izvodi se računanjem umnožaka između funkcije $f(x, y)$ koja definira prostorni položaj određenog piksela slike i w vrijednosti konvolucijske maske. Pritom mijenja vrijednost određene lokacije sumiranjem susjednih piksela. Faktor konvolucije $w(0,0)$ nalazi se na istoj lokaciji kao funkcija piksela $f(x, y)$ koji se transformira. Rad kovolucijske maske prikazan je na slici 3.4.



Slika 3.4 Prikaz rada konvolucijske maske pri linearnom prostornom filtriranju [9]

Koeficijenti filtra u linearnom prostoru prikazanom na slici 3.4 daju težinsku vrijednost određene koordinate čiji se rezultat može iskazati izrazom (3.9):

$$R = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + \ldots + w(0,0)f(x,y) + \cdots + w(1,0)f(x+1,y) + w(1,1)f(x+1,y+1) \quad (3.9)$$

Za konvolucijsku masku dimenzija $m \times n$ koja konvulira sliku dimenzija $M \times N$ primjena linearnog prostornog filtriranja provodi se prema izrazu (3.10):

$$(f \cdot w)(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} f(x+s, y+t) \cdot w(s,t), \quad (3.10)$$

Pri čemu su a i b pozitivni cijeli brojevi čija je vrijednost definirana dimenzijama konvolucijske maske prema izrazima (3.11) i (3.12):

$$a = \frac{(m-1)}{2} \tag{3.11}$$

$$b = \frac{(n-1)}{2} \tag{3.12}$$

Gaussov filtar

Tipičan primjer linearnog prostornog filtriranja je Gaussov filter koji se koristi u svrhu uklanjanja šuma te nepotrebnih detalja na slici. Jednodimenzionalna Gaussova funkcija prikazana je izrazom (3.13):

$$g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \tag{3.13}$$

Pri čemu je σ standradna devijacija distribucije koja ima ulogu kontrole stupnja zaglađenosti. Distribucija je pretpostavljena oko središnje vrijednosti 0. Standardna devijacija u Gaussovoj funkciji ima važnu ulogu u njenom ponašanju. Vrijednosti locirane između $\pm 3\sigma$ obuhvatiti će 97%. svih rezultata Ove značajke Gaussove funkcije su važne kod kreiranja Gaussove maske. Dvodimenzionalna Gaussova funkcija prikazana je u izrazu (3.14):

$$g(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{-(x^2+y^2)}{2\sigma^2}} \tag{3.14}$$

Konvolucijska maska dimenzija 3×3 kod dvodimenzionalne Gaussove funkcije, pri čemu je standardna devijacija $\sigma = 1$ dana je izrazom (3.15):

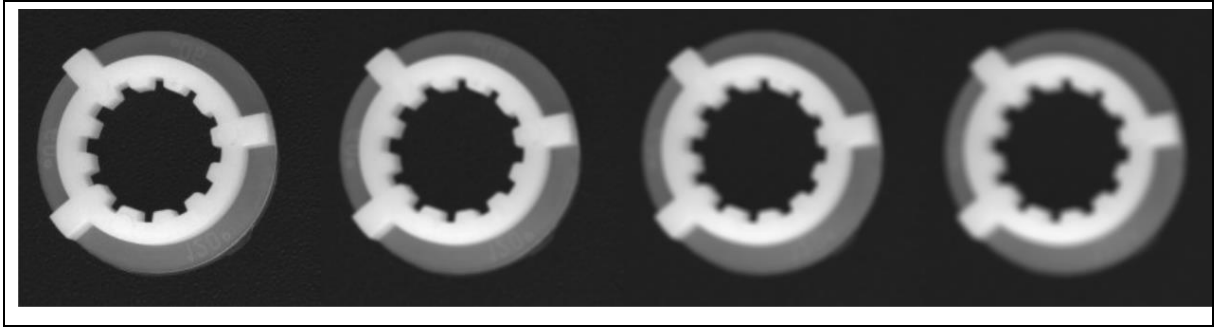
$$g(x,y) = \frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array} \tag{3.15}$$

Dok je konvolucijska maska dimenzija 3×3 pri istim uvjetima kao kod (3.15) dana izrazom (3.16)

$$g(x,y) = \frac{1}{273} \times \begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix} \quad (3.16)$$

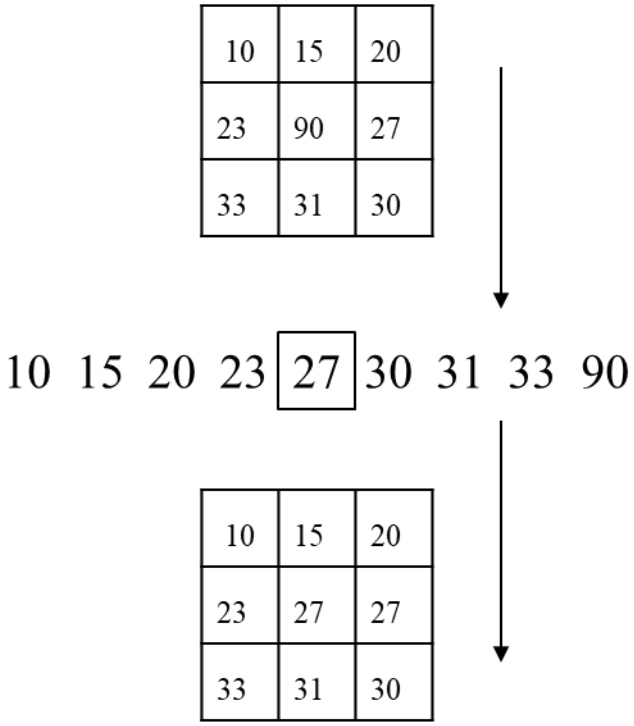
Rezultati filtriranja Gaussovog filtrom primjenom maski različitih dimenzija prikazane su u tablici 3.1.

Tablica 3.1 Rezultati filtriranja Gaussovim filtrom maskama različitih dimenzija

Bez maske	**3x3**	**5x5**	**7x7**
			

3.2.2. Nelinearno prostorno filtriranje

Nelinearno prostorno filtriranje je također bazirano na operacijama koje koriste susjedne piksele pri procesu filtriranja. Nelinearno filtriranje se temelji na operacijama koje uključuju piksele u susjedstvu obuhvaćenom filtrom. Najpoznatiji filter nelinearnog filtriranja zasniva se na traženju srednje vrijednosti funkcije distribucije kao što je prikazano na slici 3.5. Standardni medijan filtar radi na način da zamjenjuje vrijednost centralnog piksela sa srednjom vrijednosti intenziteta u njegovoj blizini. Dobre karakteristike pokazuje kod smanjenja šuma u situacijama izrazite zrnatosti slike ili kod neočekivanih šumova.



Slika 3.5 Određivanje te primjena standardnog medijan filtra

3.3. Filtriranje u frekvencijskoj domeni

Godine 1807. Fourier je iznio teoriju da se periodično ponavljanje može prezentirati kao suma sinusa i kosinusa različitih frekvencija, pri čemu se svaka množi s različitim težinama. Ova teza u svojim početcima nije bila najbolje prihvaćena. Međutim, danas, upravo ovakve tehnike pružaju značajan i praktičan način filtriranja i omogućavaju implementaciju različitih drugih tehnika obrade i analize slike [6].

3.3.1. Dvodimenzionalna diskretna Fourierova transformacija

Fourierova transformacija predstavlja signal kompleksnih eksponencijala. U dvodimenzionalnom diskretnom prostoru za funkciju *f(x,y)* gdje je digitalna slika veličine $M \times N$, dvodimenzionalna diskretna Fourierova transformacija može se pisati prema izrazu (3.17):

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{3.17}$$

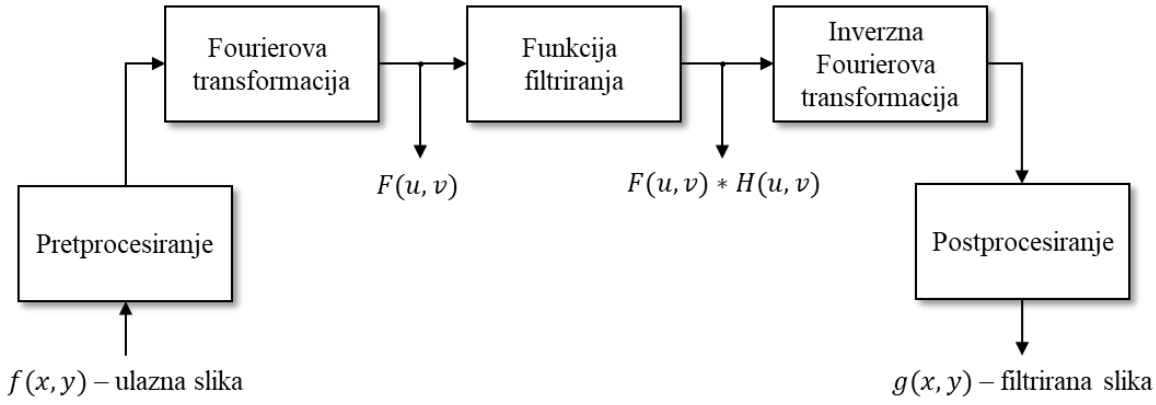
gdje je:

$F(u,v)$ – diskretna Fourierova transformacija.

Temelj filtriranja neovisno je li se radi o frekvencijskoj ili prostornoj domeni leži na teoremu konvolucije prikazane izrazom (3.18):

$$f(x,y)h(x,y) \leftrightarrow H(u,v) * F(u,v) \tag{3.18}$$

Simbol $*$ označava konvoluciju dviju funkcija, a *F(u,v)* konstantan Fourierov transformacijski par. Osnovna shema filtriranja u frekvencijskoj domeni prikazana je na slici 3.6.



Slika 3.6 Osnovna shema filtriranja u frekvencijskoj domeni [10]

3.3.2. Filtri u frekvencijskoj domeni

Niskopropusni filtar ili kraće NP (engl. *Low Pass Filter; LPF*) – izvršava zaglađivanje te smanjenje šuma slike. Prigušuje se dio spektra signala slike putem konvolucijskog signala niskopropusnog filtra. Izlazni signal dobiva se sintezom dva ulazna signala. Funkcija niskopropusnog filtra dana je prema izrazu (3.19) u kojem je prikazana vrijednost filtra D_0 koja definira frekvenciju prigušenja signala u idealnom niskopropusnom filtru:

$$H(u,v)_{\mathrm{LP}} = \begin{cases} 1 \; ako \; D(u,v) \leq D_0 \\ 0 \; ako \; D(u,v) > D_0 \end{cases} \tag{3.19}$$

Gdje vrijedi:

D_0 - frekvencija prigušenja; pozitivni broj;

$D(u,v)$ - udaljenost od točke (u,v) do centra filtra.; $D(u,v) = \sqrt{u^2 + v^2}$

Poznatiji niskopropusni filtri su Butterworthov i Gausov NP filtar.

Visoko propusni filtar ili kraće VP (engl. *High Pass Filter; HPF*) je vrsta filtra koji se koristi pri izoštravanju slike na način da prigušuje niske frekvencije, a visoke frekvencije ostavlja nepromijenjenima.

Prijenosna funkcija visokopropusnog filtra $H(u,v)_{\mathrm{HP}}$ definirana je izrazom (3.20).

$$H_{\mathrm{LP}}(u,v) = 1 - H_{\mathrm{HP}}(u,v) \tag{3.20}$$

3.4. Segmentacija

Segmentacija slike u većini slučajeva predstavlja jedan od najznačajnijih procesa u obradi slike. Ovaj proces izdvaja objekte od ostatka scene ili pozadine slike prema određenom parametru, koji može biti i značajka. Pronalaze se regije (segmenti) piksela koji pripadaju određenoj grupi prema definiranom parametru. Jednom segmentirane slike i grupirani objekti mogu ići u daljnju obradu, analizu te kasnije i razumijevanje slike. U statistici ili strojnom učenju ovaj problem je poznat pod nazivom klaster analiza te je široko primijenjen u stotinama različitih algoritama. Najveća razlika između klaster metode i segmentacije ogleda se u tom što klaster metoda ignorira raspored piksela i odnosa susjedstva, dok se segmentacija oslanja na prostorne upute i ograničenja. Kada se govori o računalnom vidu segmentacija je jedan od postupaka obrade slike koji predstavlja široko proučavan problem čiji se rad temelji na primjeni algoritama [5-6]. Ne postoji univerzalna tehnika segmentacije koja će dati željene rezultate kod svih problema niti je bilo koja tehnika savršena.

Tri su glavna pristupa procesu segmentacije:

1. Određivanje praga (engl. *Thresholding*) – pikseli su dodijeljeni u kategorije prema vrijednosti intenziteta piksela. Pikseli koji se nalaze iznad definirane granice praga pripadaju kategoriji objekta a ostali u kategoriju scene.
2. Rubno temeljena segmentacija (engl. *Edge Based*) – primjenjuje se filter koji detektira rub na temelju inteziteta piksela nakon čega izvršava klasifikaciju povezanih regija koje detektira kao rubnu konturu promatranog objekta.
3. Konačno, regijski temeljena (engl. *Region Based*) segmentacija kod koje algoritmi grupiraju ili dijele susjedne piksele ovisno o promatranom parametru.

Osim navedenih, popularne tehnike koje se koriste pri segmentaciji su: "Klaster" metoda, "Watershed" metoda, PDE metoda (temeljena na parcijalnoj derivaciji) i metoda koja primjenjuje umjetne neuronske mreže ANN. Njihove značajke dane su u tablici 3.2 [11].

Tablica 3.2 Pregled popularnih metoda segmentacije [11]

Segmentacijska tehnika	Opis	Prednosti	Nedostatci
Metoda određivanjem praga - "Thresholding"	Određuje vrijednosti praga na temelju histograma slike	Najjednostavnija metoda, nema potrebe za prethodnim pretprocesiranjem	Ovisna o vrhovima histograma (lokalnim maksimumima), prostorni detalji nisu uzeti u obzir
Metoda detekcije ruba	Bazirana na diskontinuiranoj detekciji	Povoljna za naglašavanje kontrasta između objekata	Pogrešna detekcija ruba ili prevelik broj rubova
Metoda detekcijom regije	Temelji se na podjeli slike na homogene regije	Više otporna na šumove	Skupa metoda u pogledu vremena i memorije
Klaster metoda	Temelji se na podjeli slike na homogene grupe	Zamućivanje djelomično prisutno -korisno za realne probleme	Određivanje funkcije pripadnosti nije jednostavno
"Watershed" metoda	Bazirana na topološkoj interpretaciji	Stabilniji rezultati, detektiranje granica je kontinuirano	Kompleksno izračunavanje gradijenta
PDE metoda	Bazirana na radu s različitim operacijama	Najbrža metoda, pogodna kod procesa koji zahtijevaju kraće vrijeme	Veća računska složenost
ANN metoda	Bazirana na simulaciji procesa učenja te donošenja odluka	Paralelno raspodijeljena obrada informacija	Više vremena uloženo u proces učenja

Određivanje praga

Određivanje praga istovremeno predstavlja jednostavan i snažan pristup segmentaciji slike. U računalnom vidu, tehnika određivanja praga provodi se na monokromatskim slikama. Postavljanjem praga vrijednosti t, piksel koji se nalazi na lokaciji (x, y), s vrijednosti intenziteta piksela $f(x, y)$, se grupira u prvu kategoriju koja se može definirati kao kategorija objekta prema izrazu (3.21) :

$$f(x, y) \leq \mathrm{t} \tag{3.21}$$

U slučaju da izraz (3.21) nije ispunjen, piksel je grupiran u drugu kategoriju, odnosno kategoriju scene. U brojnim slučajevima vrijednost praga t određuje se ručno, kroz testiranje područja vrijednosti t i odabira prikladne razine za objekt od interesa. Granica između kategoriziranih

piksela jasno je vidljiva u tablici 3.3, u kojoj je prikazano određivanje praga za ispitni uzorak. U konkretnom primjeru prag je postavljen na dvije različite vrijednosti intenziteta. Prag je uspješno segmentirao sliku na dvije dominantne regije od kojih je od značaja vrijednost praga od 180 obzirom da dobivena segmentacija daje najpovoljnije rezultate.

Tablica 3.3 Tri segmentacije korištenjem ručno postavljenom praga

T = 120	T = 180	T = 210
		

Binarizacija slike primjenom praga koji koristi intenzitet piksela računa se prema (3.22):

$$g(x,y) = \begin{Bmatrix} 255, if\ f(x,y) > \mathrm{T} \\ 1, if\ f(x,y) \leq \mathrm{T} \end{Bmatrix} \tag{3.22}$$

Ovakav tip segmentacije pripada metodi globalne tehnike segmentiranja slike pri kojoj se koriste opće informacija (odnosno histogram slike, globalna svojstva teksture). Vrijednost T predstavlja središnju vrijednost praga te je konstanta za cijelu sliku. Na ulaznoj digitalnoj slici $f(x,y)$ primjenom T dobiva se izlazna funkcija binarne slike $g(x,y)$.

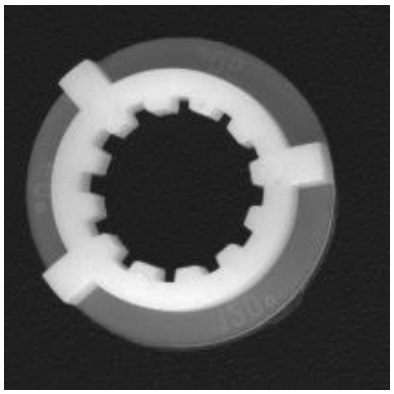
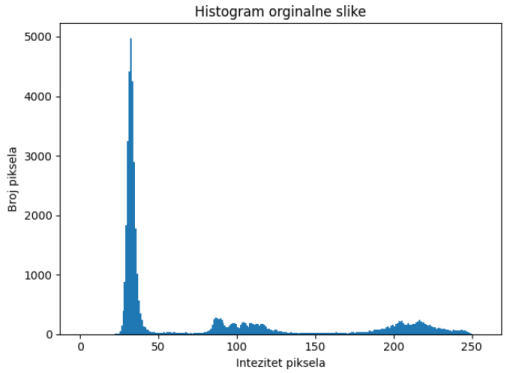
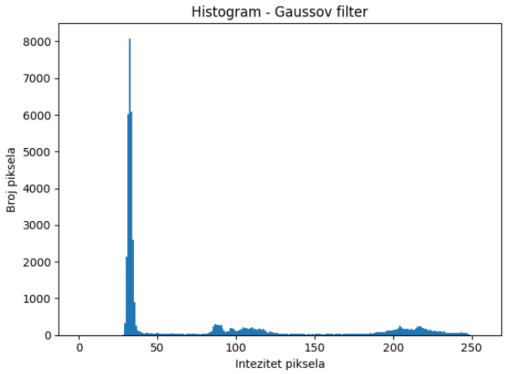
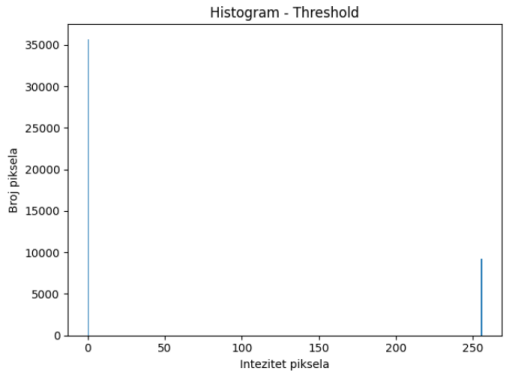
Ukoliko je vrijednost funkcije $f(x,y) > 180$, funkcija poprima vrijednost $g(x,y) = 255$ za objekte prikazane bijelom bojom te $g(x,y) = 1$ za scenu, odnosno dijelove prikazane crnom bojom kod kojih vrijedi $f(x,y) \leq 180$.

Metoda koja se zasniva na postupku određivanja praga pogodna je ukoliko objekti nisu u međusobnom dodiru i jasno se razlikuju od ostatka scene [12].

Metode koje koriste globalno određivanje praga pokazuju značajnu osjetljivost u slučajevima nejednakog osvjetljenja. Ovaj problem moguće je riješit pomoću lokanog određivanja praga pri kojem se postavljaju višestruki pragovi [13].

U primjeru u tablici 3.4 vrijednost praga definirana je ručno jer je bila pogodna za dani slučaj. Međutim ova vrijednost može biti određena i automatski [11, 14]. U tablici 3.4 prikazane su pojedinačne faze metode uz pripadne histograme.

Tablica 3.4 Faze određivanja praga

4. DIGITALNA ANALIZA SLIKE

Digitalna analiza slike (engl. *Digital Image Analysis*) je proces kod kojeg je ulazna informacija slika međutim dobivene informacije nisu slikovni podaci. Shodno tome slika se nalazi samo na ulaznom dijelu procesa (slika 4.1) [4].

Slika 4.1 Digitalna analiza slike [4]

4.1. Izdvajanje značajki

Proces izdvajanja značajki ima početnu kariku u lancu analize slike. Postupak se zasniva na predočavanjem ulaznih podataka binarne slike određenom značajkom. Značajke su obično dimenzijske vrijednosti, međutim mogu biti i tip značajki koji pokazuje intenzitet piksela, oblik i slično. Neki postupci izdvajanja značajki zahtijevaju binarnu sliku, dok ostali mogu raditi s monokromatskom slikom naglašenih rubova [2]. Prva metoda izdvajanja značajki je detekcija ruba. Detekcija ruba bavi se pronalaskom dijelova slike u kojima su izražene razlike u intenzitetu točaka [2].

4.1.1. Detekcija rubova

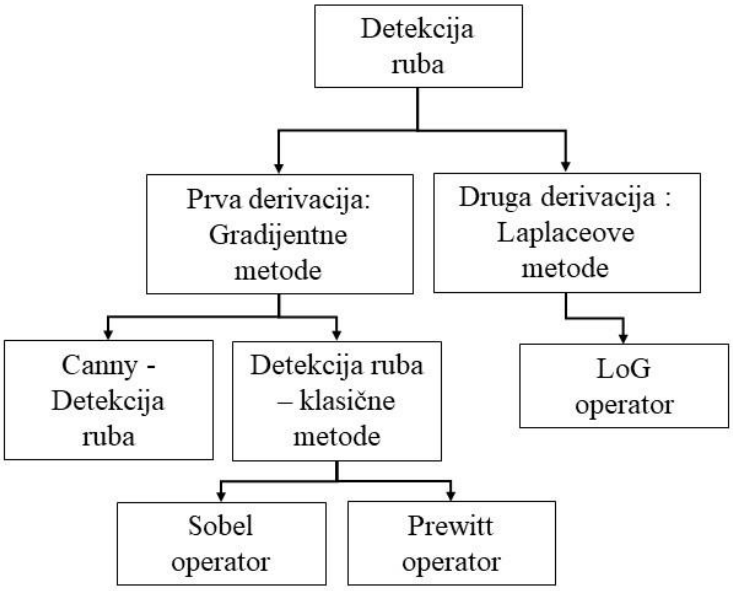
Rubovi karakteriziraju granice promatranih objekata te stoga često predstavljanju jedan od temeljnih parametara u obradi slike ili čak i analizi slike kod računalnog vida. Metoda detekcije ruba je dobro razvijena i prihvaćena u procesu obrade slike. Temelji se na praćenju nagle promjene intenziteta piksela odnosno skoka vrijednosti prilikom prelaska s jednog piksela na drugi. Pronađeno područje koristi se za detekciju i identifikaciju objekta. Detektor ruba eliminira diskontinuitet ili neovisne piksele [11, 15-16].

Metode detekcije ruba prikazane na slici 4.2 te se mogu svrstati u dvije kategorije [17]:

1. Detekcija ruba pomoću gradijentne metode:
    - Prewitt operator.
    - Sobel operator.

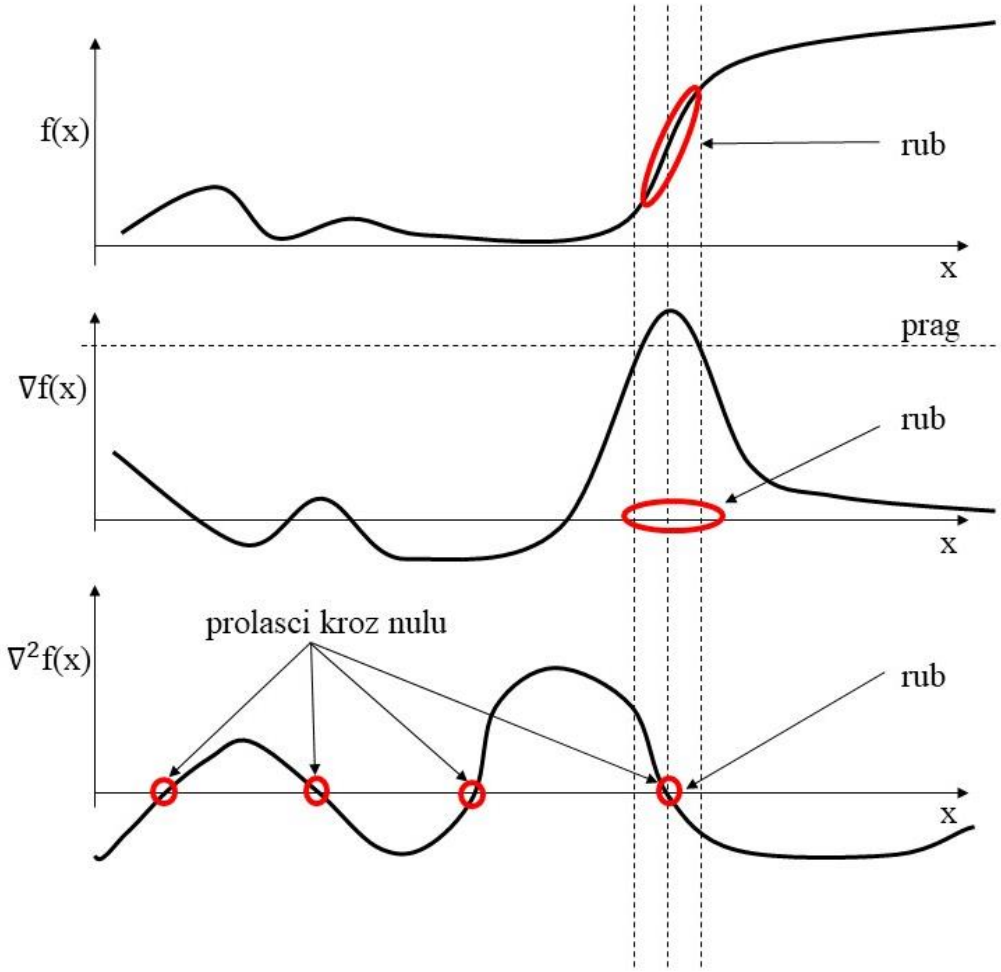
- Canny operator.

2. Detekcija ruba pomoću La placeove metode
    - Lapalceov operator.



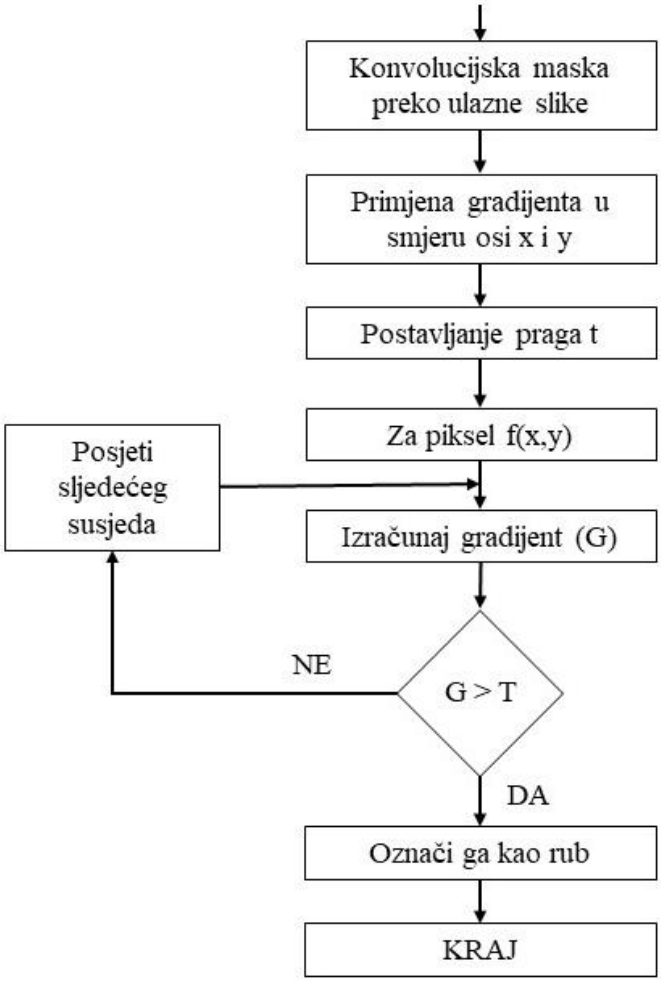
Slika 4.2 Pregled osnovnih operatora za detekciju ruba [17]

Detekcija ruba dobiva se primjenom parcijalnih derivacija na funkciju intenziteta slike kao što je prikazano na slici 4.3. Tehnike koje koriste prve derivacije funkcije intenziteta piksela pripadaju gradijentnim metodama. Točke u kojima funkcija postiže lokalni maksimum ili minimum te prelazi prag intenziteta naziva se rub. Ukoliko se provodi druga parcijalna derivacija funkcije intenziteta piksela, vrijednosti pri kojima funkcija sječe apscisu predstavlja rubno područje. Metode koje koriste drugu derivaciju funkcije opisuju se kao Laplaceove metode. Nakon postupka definiranja koordinata pojedinih točaka ruba slijedi povezivanje istih kako bi se formirala granica koja segmentira određenu regiju [15].



Slika 4.3 Prikaz ruba kod prve i druge derivacije po jednoj dimenziji digitalne slike [18]

Najpoznatiji operatori koji se koriste za detekciju rubova su Sobel, Canny, Laplace Gaussian(LoG), Prewitt i Roberts koji se razlikuju prema primjeni derivacije, lokalizacije i magnitude. Dijagram provođenja algoritma za detekciju ruba dan je na slici 4.4.



Slika 4.4 Opći tok konvencionalnog algoritma za detekciju ruba

Gradijentni postupci

Promjena funkcije slike može biti opisana gradijentom G koji pokazuje smjer gibanja najvećeg rasta funkcije slike. Rub predstavlja vektorsku varijablu s dvije komponente, smjerom i magnitudom. Magnituda ruba je magnituda gradijenta. Smjer ruba se rotira u odnosu na gradijentni smjer za -π/2. Smjer gradijenta daje smjer maksimalnog rasta funkcije f. Granica i njeni dijelovi su okomiti na smjer gradijenta [19]. Magnituda gradijenta je stoga dana u izrazima (4.1) i (4.2):

$$G[f(x,y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}^T \tag{4.1}$$

$$G[f(x,y)] = \sqrt{{G_x}^2 + {G_y}^2} \tag{4.2}$$

$$G[f(x,y)] \approx |G_x| + |G_y| \tag{4.3}$$

$$G[f(x,y)] \approx \max\left(|G_x| + |G_y|\right) \tag{4.4}$$

Iz vektorske analize smjer gradijenta je definiran kao:

$$\alpha(x,y) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{4.5}$$

Pri čemu je:

α – kut u odnosu na apscisu.

Digitalna slika je u stvarnosti diskretna te ove jednadžbe moraju biti aproksimirane primjenom raznih metoda. Najčešće se izračunava razlika gradijenata slike, u vertikalnom smjeru za fiksni *i* te u horizontalnom smjeru za fiksni *j* prema izrazima (4.6) i (4.7):

$$G_i \cong f[i, j+1] - f[i,j] \tag{4.6}$$

$$G_j \cong f[i,j] - f[i+1, j \tag{4.7}$$

Gradijenti operatori mogu biti predstavljeni konvolucijskom maskom dimenzija 2×2 prema izrazima (4.8) i (4.9)

$$G_x = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \tag{4.8}$$

$$G_y = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \tag{4.9}$$

Pri čemu je:

Gx – aproksimacija gradijenta u smjeru apscise

Gy – aproksimacija gradijenta u smjeru ordinate.

Smjer gradijenta uvijek je okomit na smjer ruba. Promatrana točka leži između sva 4 piksela u 2×2 susjedstvu interpolirane točke $\left[i + \frac{1}{2}, j + \frac{1}{2}\right]$. Ovaj faktor može voditi do konfuzije stoga je alternativna varijanta korištenje 3×3 susjedstva i računanje gradijenta oko centralnog piksela.

Najpoznatije konvolucijske maske ovih dimenzija su Sobel i Prewit čiji je rad predstavljen u nastavku.

Sobel operator

Operator izračunava dvodimenzionalni gradijent pomoću konvolucijske matrice veličine 3×3. Algoritam se bazira na prvoj parcijalnoj derivaciji funkcije, analizi derivacije i računanju vrijednosti gradijenta intenziteta slike na svakoj točki. Nakon tog daje smjer kako bi povećao intenzitet slike na svakoj točki iz svijetle u tamnu ili obrnuto. Po završetku detektira točke gdje gradijent postiže maksimalne vrijednosti[19]. Magnituda Sobelova gradijenta računa se prema izrazu (4.10):

$$M = \sqrt{s_x^2 + s_y^2} \tag{4.10}$$

Gdje se parcijalne derivacije računaju prema izrazima (4.11) i (4.12):

$$s_x = (a_2 + ca_3 + a_4) - (a_0 + ca_7 + a_6) \tag{4.11}$$

$$s_y = (a_0 + ca_1 + a_2) - (a_6 + ca_5 + a_4) \tag{4.12}$$

Sa kontanstnom c iznosa $c = 2$, i vrijednostima susjednih piksela definiranim prema (4.13):

a_0	a_1	a_2
a_7	[i,j]	a_3
a_6	a_5	a_4

(4.13)

Gradijentne aproksimacije s_x i s_y mogu biti implementirane putem konvolucijskih maski prema izrazima (4.14) i (4.15):

$$s_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{4.14}$$

$$s_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{4.15}$$

Važno je napomenuti kako ovaj operator postavlja naglasak na piksele koji su bliže centru maske. Vrijednosti iskazane u tablici 4.1 na kraju ovog poglavlja prezentiraju rezultate pojedinih operatora. Sobel operator je jedan od najviše korištenih detektora ruba [16, 19].

Prewitt operator

Prewit operator se također zasniva na primjeni prve parcijalne derivacije funkcije. Koristi masku dimenzija 3×3 za pronalaženje vrhova magnitude gradijenta. Za razliku od Sobelovog operatora kod Previtta vrijednost konstante c iznosi $c = 2$. Kada najviša vrijednost magnitude biva pronađena, operator detektira rub horizontalnom ili vertikalnom aproksimacijom gradijenta, ili kombinacijom spomenutih[16, 19].

Konvolucijske maske za gradijentne aproksimacije s_x i s_y kod Prewittovog operatora dane su izrazima (4.16) i (4.17):

$$s_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \tag{4.16}$$

$$s_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \tag{4.17}$$

Za razliku od Sobel operatora, Prewitt operator ne naglašava piksele koji su najbliži centru maske. Rad ovog detektora također je dan u tablici 4.1.

Canny operator

Canny operator predstavlja vjerojatno najpopularniju tehniku za pronalazak ruba. Razvijen je 1986. godine od strane Johna F. Cannya. Ovaj algoritam reducira količinu podataka odnosno smanjuje šum koji se generira kod detekcije ruba bolje od ostalih prethodno predstavljenih operatora. U pogledu vremena procesiranja on je računski intenzivniji u odnosu na ostale. Pripada skupini gradijentnih postupaka, odnosno njegov rad se zasniva na primjeni prve parcijalne derivacije [19, 21]. Obzirom da je Canny operator korišten pri rješavanju diplomskog zadatka, zbog rezultata koji su izneseni u narednom poglavlju, njegov rad je opisan detaljnije od ostalih predstavljenih operatora za detekciju rubova.

Canny operator koristi pet procesa kako bi postigao virtualnu granicu ispitivane slike. Izvođene operacije su redom:

1. Gaussov filtar koji se primjenjuje na sliku s ciljem zaglađivanja.
2. Pronalazi se magnituda gradijenta intenziteta slike i orijentacija pri čemu se koristi razlika aproksimacija za parcijalne derivacije.
3. Primjenjuje se *"Non-maximum suppression"* NMS – svaka vrijednost koja nije lokalni maksimum se postavlja na 0.
4. Primjenjuje se određivanje praga kako bi se pronašli potencijalni rubovi.

5. U konačnosti, rubovi se prate histerezom.

Prednosti Canny algoritma su neosjetljivost za fine geometrijske i fotometričke transformacije, dobri rezultati lokalizacije, izdvajanje značajki bez promjene istih te manja osjetljivost na šum [15-16, 20].

Prvi korak pri detekciji ruba Cannyjevim operatorom iskazan je prema (4.18). Funkcija $I[i, j]$ notira koordinate slike. Rezultat primjene Gaussova filtra je zaglađeno područje $S[i, j]$[19]:

$$S[i,j] = G[i,j;\sigma] \cdot I[i,j]. \tag{4.18}$$

Pri čemu je:

σ – širina Gaussa i kontrola stupnja zaglađivanja.

Gradijent zaglađenog područja $S[i, j]$ može biti izračunat koristeći matricu 2×2 pri izračunu prve derivacije kako bi dobili vrijednosti aproksimacije polja $P[i,j]$ i Q[i,j] za x i y parcijalne derivacije prema izrazima (4.19) i (4.20):

$$P[i,j] \approx (S[i,j+1] - S[i,j] + S[i+1,j+1] - S[i+1,j])/2 \tag{4.19}$$

$$Q[i,j] \approx (S[i,j] - S[i+1,j] + S[i,j+1] - S[i+1,j+1])/2 \tag{4.20}$$

Konačne razlike su procijenjene preko kvadratne matrice 2×2 pri čemu x i y predstavljaju parcijalne derivacije izračunate u istoj točki slike. Magnituda i orijentacija gradijenta može biti izračunata standardnom formulom za interpolaciju kvadratnih u polarne koordinate prema izrazima (4.21) i (4.22)[19]:

$$M[i,j] = \sqrt{P[i,j]^2 + Q[i,j]^2} \tag{4.21}$$

$$\theta[i,j] = \arctan(Q[i,j], P[i,j]) \tag{4.22}$$

Gdje arctan funkcija uzima oba argumenta Q i P te generira kut mogućeg smjera. Time se izbjegava pogreška do koje dolazi kod generiranja magnitude u smjeru apscise koja je jednaka 0. Smjer ruba će u tom slučaju biti jednak odmaku od 90° u odnosu na apscisu, odnosno u smjeru osi ordinate. U slučaju da je gradijent u smjeru ordinate jednak nuli, smjer ruba bit će u smjeru osi apscise [19, 21]. Sljedeći korak je povezivanje smjera ruba sa smjerom gradijenta koji je već poznat.

Magnituda slike $M[i, j]$ će imati veće vrijednosti na lokacijama gdje je gradijent slike velik. U svrhu identifikacije ruba, vrijednosti intenziteta točaka pojedinih rubova u magnitudi moraju biti smanjeni kako bi se zadržale samo točke najviših maksimuma. Ovaj proces je nazvan *non-*

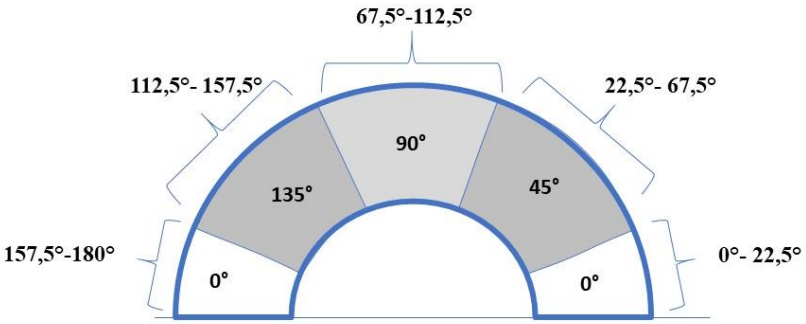
maxima suppression – NMS koji kao rezultat daje stanjene rubove. Algoritam počinje smanjivati kut gradijenta $\theta[i,j]$ na vrijednost jednog sektora $\zeta[i,j]$ od četiri moguća, čiji prikaz se nalazi na slici 4.5 unutar izraza (4.23)[16]:

$$\zeta[i,j] = Sector(\theta[i,j]) \tag{4.23}$$

Algoritam koristi matricu dimenzija 3×3 preko magnitude $M[i,j]$. Na svakoj točki, centar elementa $M[i,j]$ susjedstva je uspoređen sa dva susjedna piksela uzduž linije gradijenta dane sektorske vrijednosti $\zeta[i,j]$ na centru susjedstva. Ukoliko magnituda $M[i,j]$ u centru nije veća od obje susjedske magnitude uzduž gradijentne linije, tada joj se dodjeljuje vrijednost 0. Provedbom prezentirane metode dobivamo rubove debljine samo jednog piksela [19].

Centrirani piksel može biti iskazan u četiri smjera ovisno kojem smjeru je smjer magnitude najbliži pri opisu okolnih piksela:

- 0° - za horizontalni smjer
- 45° - uzduž pozitivne dijagonale
- 90° - za vertikalni smjer
- 135° - uzduž negativne dijagonale.



Slika 4.5 Određivanje orijentacije ruba Cannyevim postupkom [21]

Stoga će se svakoj vrijednosti ruba koji je orijentiran u rasponu vrijednosti koji pripada bijelom području [0-22,5] i [157,5-180] dodijeliti vrijednost 0.

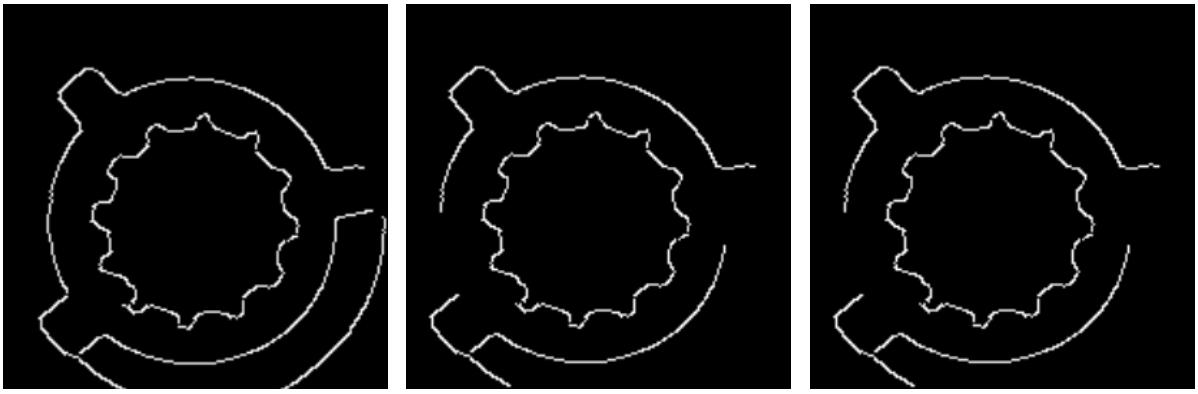
Vrijednosti za visinu intenziteta ruba dobivene su preko NMS magnitude prema izrazu (4.24):

$$N[i,j] = nms(M[i,j], \zeta[i,j]) \tag{4.24}$$

Usprkos zaglađivanju izvršenom u prvom koraku, nakon primjene NMS slike funkcija $N[i,j]$ će sadržavati mnoge lažne bridove i fragmente ruba uzrokovane šumom ili finom teksturom. Pritom je važno naglasiti kako kontrast kod lažnih rubova ima malu vrijednost [16].

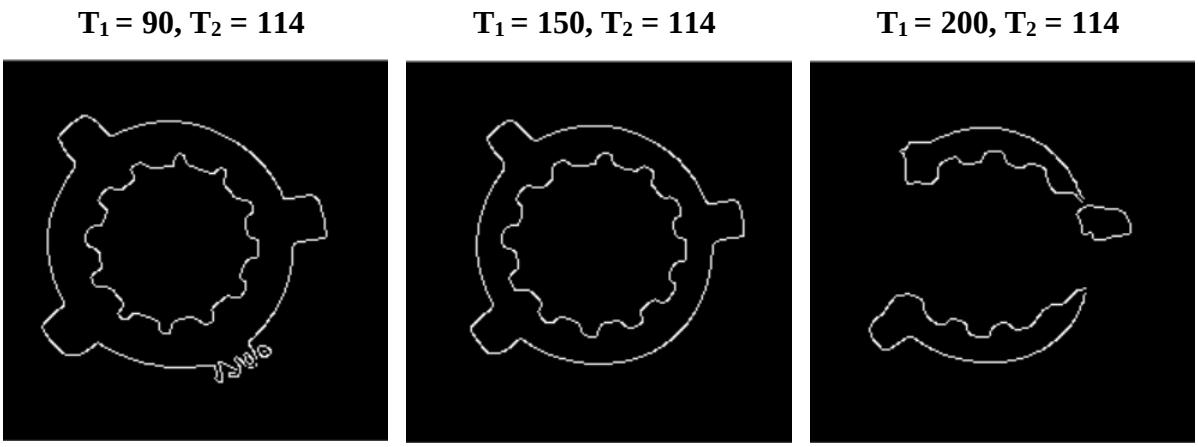
Tipični postupak uklanjanja lažnih rubova je primjena praga T na vrijednost funkcije $N[i,j]$. Sve vrijednosti koje se nalaze ispod praga su postavljene na 0. Korištenjem različitih vrijednosti postavljenog praga dan prikazano je na slici 4.6.

$\boldsymbol{T = 90}$ $\boldsymbol{T = 150}$ $\boldsymbol{T = 150}$



Slika 4.6 Detekcija ruba primjenom Canny operatora za različite vrijednosti praga

Problemi koji se eventualno mogu pojaviti kod ove faze Cannyevog algoritma je vrijednost praga čije pogrešno postavljanje može dati diskontinuitet konture ili prepoznavati nepotrebne šumove poput bridova. Jedan od načina postizanja zadovoljavajućeg praga je postavljanje dvaju vrijednosti praga, odnosno primjena histereze. Dvostruki prag uzima $N[i,j]$ i primjenjuje prag t_1 i t_2 , pri čemu je $t_2 = 2 \times t_1$, te dobiva dvije slike $T_1[i,j]$ i $T_2[i,j]$. Obzirom da je T_2 slika formirana s višim pragom, sadržavat će manji broj lažno detektiranih bridova, međutim moguća je pojava diskontinuiteta konture. Stoga će dvostruki prag u algoritmu koristiti za povezivanje konture u T_2. Kad dosegne kraj konture algoritam gleda u lokacije u T_1 u području 8 susjedstva za rub koji može biti spojen u konturu. Algoritam nastavlja skupljati bridove povezujući ga kao umnožak pragova i rješava probleme pravilnog odabira praga [19, 21]. Korištenje dvostrukog praga prikazan je na slici 4.7.



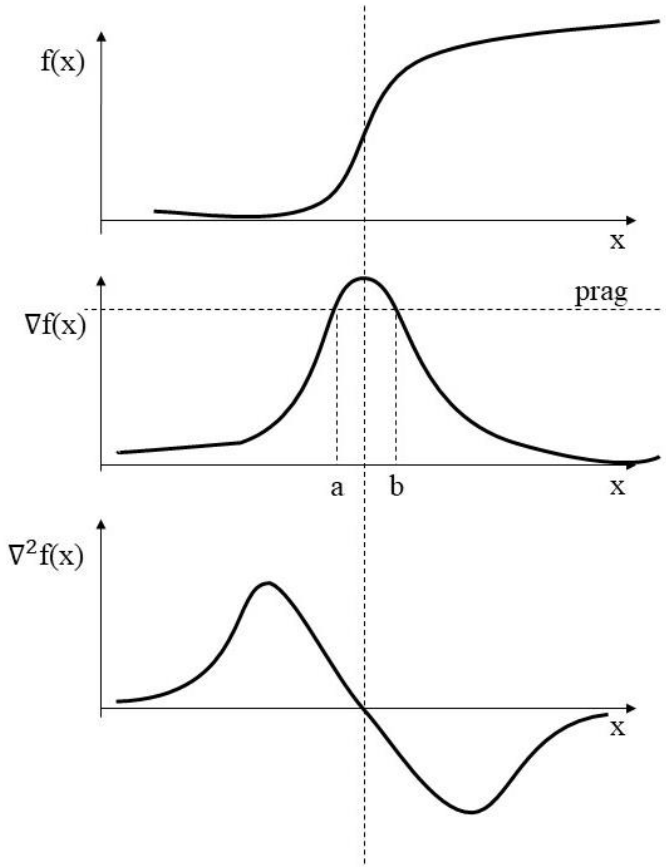
Slika 4.7 Detekcija ruba temeljena na Canny operatoru primjenom dvostrukog praga

Laplaceovi postupci

Prethodno predstavljeni postupci detekcije ruba dobiveni su korištenjem prve parcijalne derivacije funkcije slike. Ukoliko je funkcija prelazila vrijednosti praga, pretpostavila se točka ruba. Međutim, ovaj postupak može dati rezultate koji imaju veći broj točaka ruba odnosno detektirat će sve točke koje se nalaze u rasponu $[a, b]$ kako je prikazano na slici 4.8. Bolji pristup je pronalazak lokalnih maksimuma nakon provedene prve parcijalne derivacije funkcije. Točke u kojima je postignut lokalni maksimum smatrati će se točkama ruba. Obzirom na ponašanje funkcije da u zonama rasta odnosno pada ostvaruje lokalne maksimume i minimume, druga derivacija će na istima dati vrijednosti 0. Stoga se druga derivacija također može koristiti kao alat detekcije ruba [16, 19]. Tipičan primjer ove detekcije je Laplaceov operator.

Laplace Gaussian operator, ili kraće LoG (engl. *Laplacian of Gaussian*) zasniva se na principu druge parcijalne derivacije pri kojoj vrijednost funkcije $f(x, y)$ slike presijeca apscisu kao što je prikazano na slici 4.8. Laplaceov operator je stoga dvodimenzionalni ekvivalent druge derivacije funkcije. Formula za Laplace funkciju $f(x, y)$ dana je izrazom (4.25):

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{4.25}$$



Slika 4.8 Detekcija ruba primjenom Laplaceove metode [19]

Laplaceov operator signalizira prisutnost ruba kada izlazni parametri operatora naprave prijelaz polariteta. Ako bi ulazna slika imala vrijednosti intenziteta piksela dane na slici 4.9 izlazni odziv Laplaceovog operatora bi dao vrijednosti prikazane slikom 4.10.

2	2	2	2	8	8	8	8
2	2	2	2	8	8	8	8
2	2	2	2	8	8	8	8
2	2	2	2	8	8	8	8
2	2	2	2	8	8	8	8

Slika 4.9 Ulazna slika s naznačenim vrijednostima piksela [19]

0	0	0	6	-6	0	0	0
0	0	0	6	-6	0	0	0
0	0	0	6	-6	0	0	0

Slika 4.10 Izlazni Laplaceov odziv [19]

Prijelaz piksela iz pozitivne u negativnu vrijednost direktno korespondira prijelazu intenziteta piksela na slici. Odabrani primjer je ilustrativne prirode kod idealnih situacija, te ga ne možemo primijeniti kad se radi o realnoj slici kod koje imamo pojavu šuma.

Ususret rješavanju ovog problema 1980. godine izašli su Marr i Hildreth. U radu objavljenom pod nazivom: *"Teorija detekcije ruba"* (engl. *"Therory of Edge detection"*), predložili su kombinaciju Gaussovog filtra sa Laplaceovom detekcijom ruba. Primjenom Gaussovog filtra reducira se šum i diskontinuiteti. Obzirom da će zaglađivanje povećati širinu ruba, rub će se smatrati samo vrijednostima postignutog maksimum pri provedbi prve derivacije funkcije, nakon čega se primjenom druge derivacije traže nul točke[19]. Izlazni parametri su nakon primjene ovog operatora $h(x, y)$ dobiveni konvolucijskom operacijom prema izrazu (4.26):

$$h(x, y) = \nabla^2[(g(x, y) * f(x, y)] \tag{4.26}$$

kod koje se koristi derivacijsko pravilo konvolucije prema (4.27)

$$h(x, y) = [\nabla^2(g(x, y) * f(x, y)] \tag{4.27}$$

pri čemu je Laplaceov operator dan prema izrazu (4.28):

$$\nabla^2 g(x, y) = \left(\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4}\right) e^{-\frac{(x^2+y^2)}{2\sigma^2}}, \tag{4.28}$$

Veličina redukcije šuma u ovisnosti je o σ. Što je veći σ bolje je filtriranje uz neznatan gubitak važnih informacija. Ukoliko se koristi mali filtar, šum je velik. Kod primjene velikog filtera, rubovi koji se nalaze jedni blizu drugih mogu biti spojeni te detektirani kao jedan rub. Ispravna veličina filtera ne može biti utvrđena bez informacije o veličini promatranog objekta na sceni[16, 19].

4.1.2. Metodologija određivanja metode detekcije ruba

Evaluacija spomenutih postupaka detekcije ruba napravljena je promatranjem vrijednosti ispravno detektiranog ruba, vremena procesiranja, ranga pogreške te razine šuma. U obzir su uzeti Canny, LoG, Sobel i Prewit operatori.

Ispitivanje rezultata je provedeno na način da je prvo primijenjen željeni operator korištenjem algoritma razvijenog u programskom jeziku Python. Zatim je provedeno mjerenje vrijednosti vršnog omjera signala i šuma PSNR (engl. *Peak Signal to Noise Ratio*) i srednje kvadratne pogreške MSE(engl. *Mean Squared Error*) za rezultante slike sa detektiranim rubom.

Srednja kvadratna pogreška

Srednja kvadratna pogreška ili kraće MSE uključuje degradaciju funkcije i statističku karakteristiku šuma u slici kod koje je detektiran rub. MSE uspoređuje prosječnu razliku piksela temeljenu na izvornoj slici sa slikom na kojoj je detektiran rub. Viša vrijednost MSE indicira veću razinu između prvobitne i procesirane slike prema izrazu (4.29)

$$MSE = \sum_{M,N} \frac{[I_1(m,n) - I_2(m,n)]^2}{M,N} \tag{4.29}$$

Pri čemu je:

I_1 - funkcija originalne slike

I_2 - slika na kojoj je detektiran rub

m,n - visina i širina promatrane slike.

U pogledu rekonstrukcije ili kompresije slike poželjno je da vrijednost MSE bude što manja. Međutim, kada se govori o detekciji rubova, MSE treba biti što veći kako bi jamčio da je pronađen broj točaka sposoban detektirati konture te da ignorira područja slike koja ne pripadaju rubnim dijelovima[16].

Vrijednost vršnog omjera signala i šuma

Vrijednost vršnog omjera signala i šuma PSNR predstavlja raspon između maksimalne moguće snage signala i snage skupljenog šuma koji utječe na vjernost njegove reprezentacije. PSNR se obično izražava u decibelima. Slično kao MSE, kod postupaka koji se bave kvalitetom slike poželjno je da je vrijednost ovog parametra što veća, međutim u slučaju detekcije ruba, PSNR mora biti što manji kako bi se dobili željeni rezultati [16]. PSNR se izračunava na temelju poznatog MSE prema izrazu (4.30):

$$PSNR = 10log_{10}\left(\frac{R^2}{MSE}\right) \tag{4.30}$$

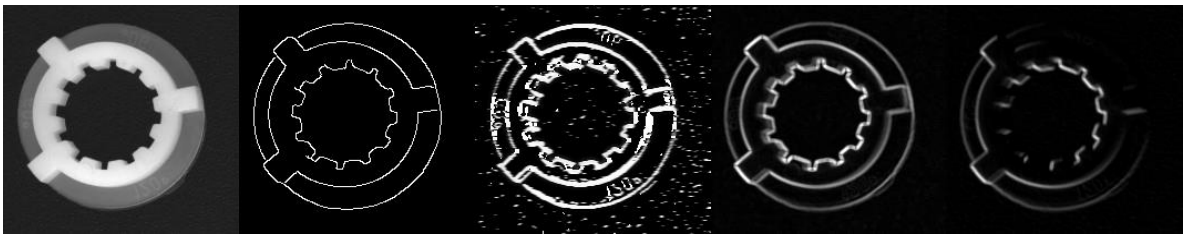
Pri čemu je:

R - maksimalni raspon ulaznih podataka slike (kod 8 bitnih registara, $R = 255$)[16].

Rezultati ispitivanja metode detekcije ruba

Uspoređena su četiri različita detektora ruba: Canny, LoG, Sobel i Prewit. Ispitivanje se temelji na procjeni rada operatora prema dobivenim vrijednostima MSU i PSNR. Slika objekta na kojoj je izvršena analiza je monokromatska i ima dimenzije 214×214. U prvom koraku, prije primjene detekcije ruba, izvršeno je zaglađivanje primjenom Gaussovog filtra ranije predstavljenog. Primjeri su testirani korištenjem algoritma razvijenog u programskom jeziku Python. Rad operatora kod detekcije rubova prikazan je na slici 4.11.

Ulazna slika	Canny operator	LoG operator, maska 5×5	Sobel operator u oba smjera, maska 3×3	Prewit, operator u oba smjera, maska 3×3



Slika 4.11 Prikaz rada različitih operatora pri detekciji ruba

Za prethodno prikazanu detekciju rubova, usporedbu metoda možemo izvršiti praćenjem parametara (primjerice kontinuiteta ruba, veličine šuma, vrijeme procesiranja i slično). Pri mjerenju šuma korišteni su MSE i PSNR čiji su rezultati prezentirani u tablici 4.1. Spomenute metode su se koristile u svrhu procjene kvalitete detekcije ruba. MSE prezentira kumulativnu pogrešku između detektiranog ruba i izvorne ulazne slike dok PSNR prezentira mjeru pogreške u vrhovima rubova. U usporedbi sa ulaznom slikom ukoliko operator daje vrijednost s manjim PSNR i višim MSE tada se dolazi do zaključka da ovaj operator ima veću sposobnost detekcije ruba. Vrijednost šuma mora biti što manja kako bi se dobio efektivan rub na detektiranoj slici. Pri tom se ističe Canny metoda detekcije koja daje 21,818 dB. U usporedi s preostala tri filtra, Canny operator smanjuje šum za 6 dB. Pri razini MSE izdvaja se Sobel operator koji postiže

vrijednost 111,89. Prema vremenu procesiranja Prewitt operator pokazuje bolje rezultate od ostalih operatora. Pri odabiru operatora napravljena je i vizualna kontrola koristeći sliku 4.11[16].

Tablica 4.1 Usporedba metoda detekcije ruba

		Canny	LoG	Sobel	Prewitt
Osnovna slika	MSE	107,46	106,86	111,89	109,25
	PSNR, dB	21,818	27,84	27,64	27,74
	Vrijeme procesiranja	0,0040s	0,0080 s	0,0030	0,0026

Rezultati dobiveni pri usporedbi postojećih metoda korišteni su u eksperimentalnom dijelu rada.

4.2. Postupci izdvajanja značajki temeljen na Houghovim transformacijama

Godine 1962, Paul Hough predlaže metode i sredstva za raspoznavanje kompleksnih uzoraka (engl. *Methods and Means for Recognition Complex Patterns*) na slici ili drugoj slikovnoj prezentaciji. Ovaj patent poznat je pod nazivom Houghova transformacija [21].

Houghova transformacija (HT) je algoritam koji određuje parametre značajki određenog oblika na slici, te ih izolira. Oblici kod kojih se primjenjuje moraju zadovoljavati određene geometrijske značajke. Stoga se koristi samo kod detekcije linija, krugova i elipsi. Ovaj algoritam doživljava konstantno poboljšanje zbog svoje široke primijenjenosti. U usporedbi s ostalim metodama koje se koriste za prepoznavanje oblika (primjerice *Template Matching*), Houghova transformacija je računski jednostavnija međutim, i dalje je intenzivna.

Prije provođenja postupka Houghove transformacije, ulazna slika mora biti obrađena, filtrirana. i binarizirana sa izraženim rubovima. Binarizacija se obično izvršava jednostavnom primjenom praga, nakon koje se primjenjuje Canny algoritam, čime se završava postupak definicije detekcije ruba. [21].

Transformacija se potom izvodi prebacivanjem iz izvornog u parametarski prostor značajki, nazvan Houghov prostor ili akumulator, u kojem su parametri definirani kao konstante čija vrijednost akumulacije ukazuje na postajanje objekta [21].

4.2.1. Metoda za detekciju pravaca

Kao osnova postupka transformacije prezentirana je Houghova transformacija za pravce. Pravac je definiran skupom točaka koje se na njemu nalaze te parametrima (*k*,*l*) prema izrazu (4.31):

$$y_i = kx_i + l \tag{4.31}$$

Pri čemu vrijedi:

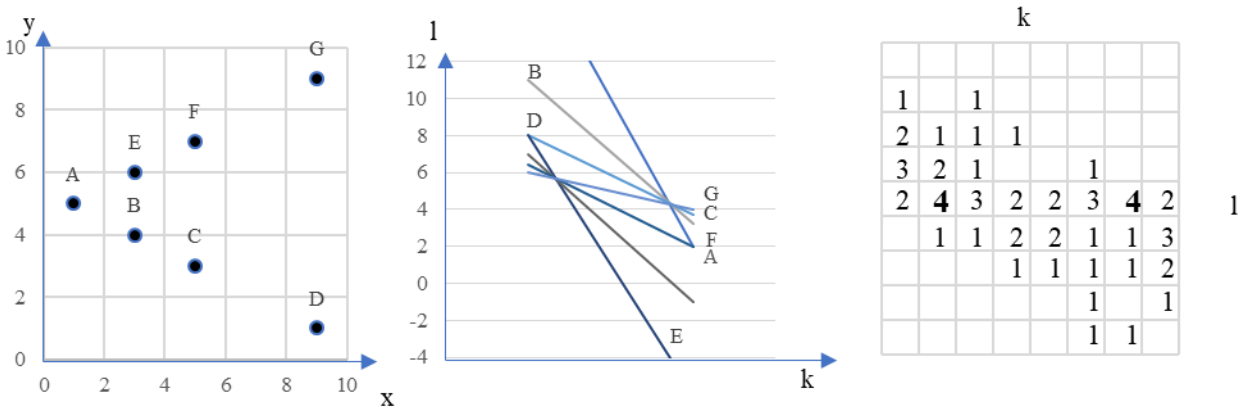
k – koeficijent smjera pravca

l – odsječak na ordinati.

Iz navedene definicije pravca preformulacijom dobivamo izraz (4.32):

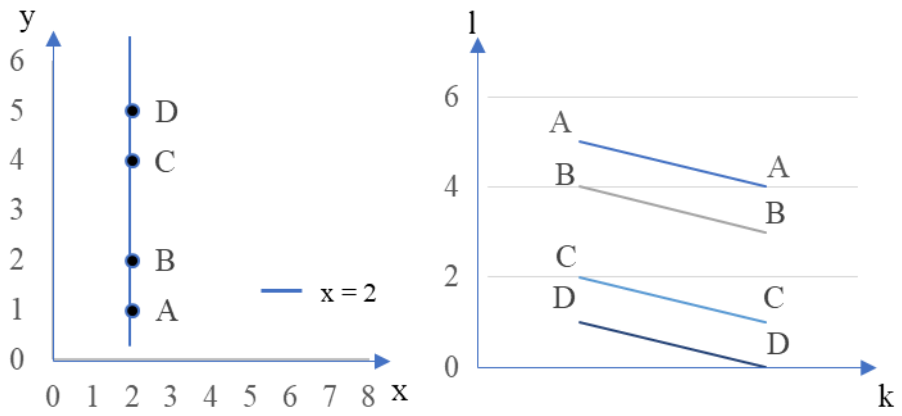
$$l = y_i - kx_i \tag{4.32}$$

Prema navedenom izrazu svaka točka u akumulatoru transformira se u pravac koji ima pripadajuće koordinate (*k*,*l*), koeficijent smjera definiran sa x_i te odsječak na ordinati definiran s y_i. Obzirom na nove podatke kreira se akumulator koji se povećava svakim dodatnim dodjeljivanjem točke sa pripadajućim (*k*,*l*) koordinatama. Na slici 4.12 prikazana su dva takva sjecišta. Kod promatranja različitih točaka u transformiranom prostoru dolazi do promjene vrijednosti x_i i y_i uz konstantne (*k*,*l*) . U transformiranom parametarskom prostoru jasno se daju iščitati guste točke. Dobivene vrijednosti *k* i *l* se odabiru kao parametri pravca[22].



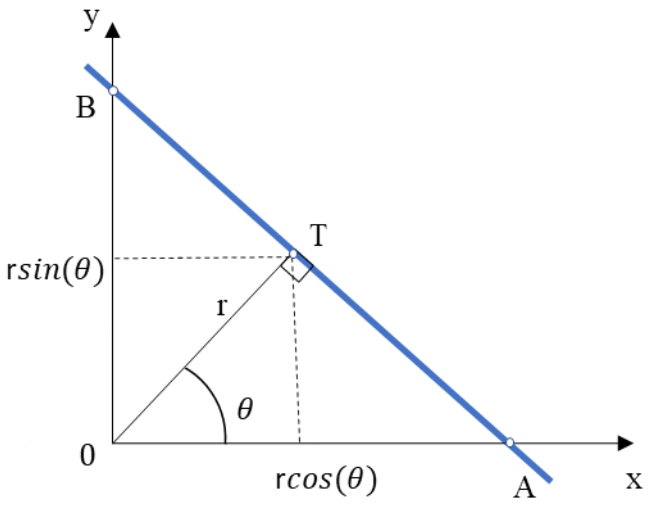
Slika 4.12 Prikaz Houghove transformacije za pravce [22]

Nedostatak ovog postupka je vrijednost varijable *k* koja može varirati $\pm\infty$ u slučaju kad se radi o vertikalnim linijama prema slici 4.13.



Slika 4.13 Houghova transformacija vertikalnog pravca [22]

Richard Duda i Peter Hart su 1972. godine objavili članak u kojem koriste takozvanu normalnu parametrizaciju za pravce. Normalna parametrizacija rješava problem koji se javlja kod vertikalnih pravaca transformacijom u polarne koordinate kao što je prikazano na slici 4.14[22].



Slika 4.14 Normala na liniju u koordinatnom sustavu slike [22]

Normala izlazi iz ishodišta 0, koje odgovara vrijednosti minimalne udaljenosti pravca od ishodišta. Ova udaljenost je dana s parametrom r i kutom u odnosu na horizontalu θ prema sljedećim izrazima:

$$r = OD \tag{4.33}$$

$$\cos\theta = \frac{r}{\overline{OA}} \tag{4.34}$$

$$\sin\theta = \frac{r}{\overline{OB}} \tag{4.35}$$

$$OA = \frac{r}{cos\theta} \tag{4.36}$$

$$OB = \frac{r}{sin\theta} \tag{4.37}$$

(4.38)

Ako bi jednadžbu pravca iz (4.31) modificirali na način prikazan u (4.39) i (4.40):

$$k = -\frac{l}{m} \tag{4.39}$$

$$\frac{y}{l} - \frac{kx}{l} = 1 \tag{4.40}$$

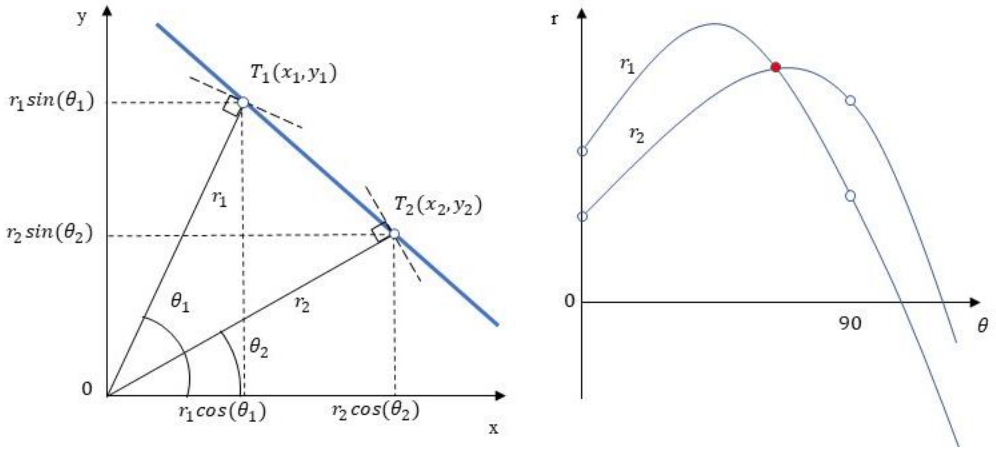
Pri čemu vrijedi $m = OA$, a $l = OB$, dolazimo do izraza prikazanih u (4.41), (4.42) i (4.43):

$$\frac{x}{\frac{r}{\cos\theta}} + \frac{y}{\frac{r}{\sin\theta}} = 1 \tag{4.41}$$

$$r = x\cos\theta + y\sin\theta \tag{4.42}$$

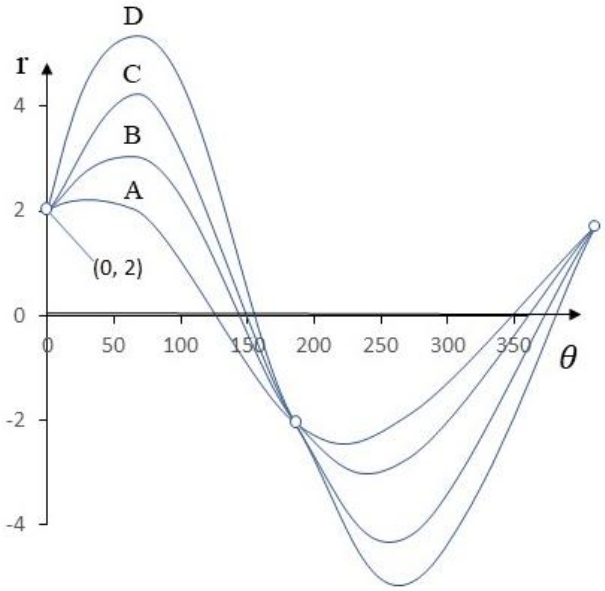
$$y = r\sin^{-1}\theta - x\mathrm{tg}^{-1}\theta \tag{4.43}$$

Pravac s normaliziranim parametrima će dati vrijednosti u transformiranom koordinatnom sustavu te seriju periodičkih krivulja u polarnom koordinatnom sustavu prikazanu na slici 4.15 [22].



Slika 4.15 Primjer Houghove transformacije za pravce s polarnim koordinatama [22]

Za primjer dan na slici 4.13 dana je transformacija u polarne koordinate prikazana na slikama 4.16 i 4.17.



Slika 4.16 Houghova transformacija vertikalnog pravca u polarnim koordinatama [22]

Ukoliko se značajka detektira unutar kruga čiji se centar nalazi u ishodištu koordinatnog sustava, koristi se cirkularna funkcija prema izrazu (4.44):

$$r^2 = x^2 + y^2 \tag{4.44}$$

U slučaju da je krug pomaknut od ishodišta za vrijednosti a i b, opća jednadžba može se prikazati izrazom (4.45):

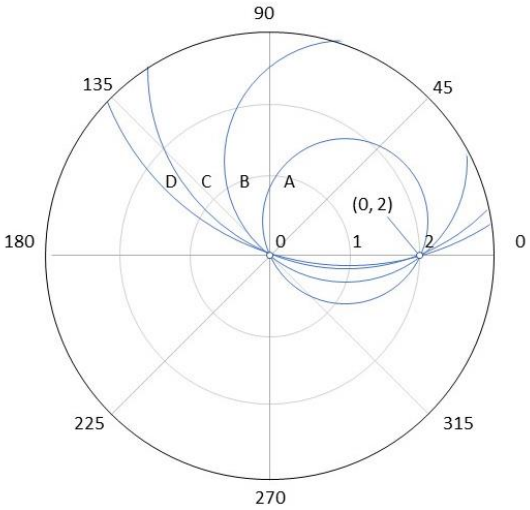
$$r^2 = (x - a)^2 + (y - b)^2 \tag{4.45}$$

Pri čemu je:

r – radijus kruga

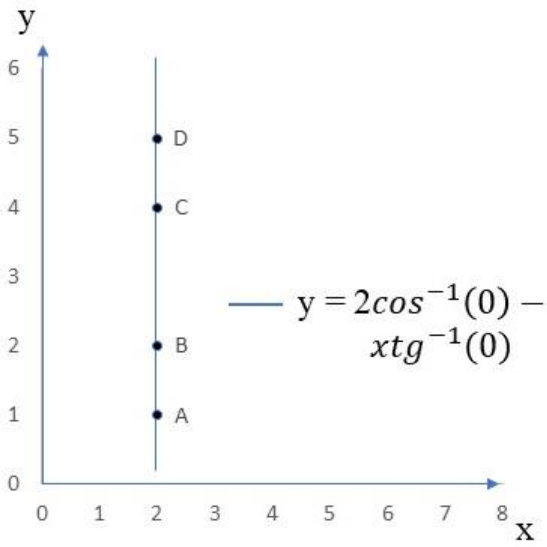
a – udaljenost od ishodišta u smjeru apscise

b – udaljenost od ishodišta u smjeru ordinate.



Slika 4.17 Prikaz Houghove transformacije za tražene točke na vertikalnom pravcu [22]

Primjenom Houghove transformacije u polarnim koordinatama detektiran je pravac prikazan na slici 4.18.

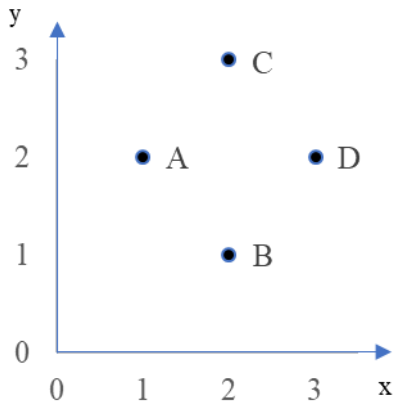


Slika 4.18 Detekcija vertikalnog pravca primjenom Houghove transformacije u polarnim koordinatama [22]

4.2.2. Metoda za detekciju krugova

Houghova transformacija se osim za linije koristi i pri detekciji elipsi i krugova. Kružna Houghova transformacija pronalazi radijus te centar promatranog objekta na sceni. Na slici 4.19

prikazane su četiri točke koje su položene na imaginarnoj kružnici čije postojanje u ovom trenutku nije evidentno.



Slika 4.19 Prikaz točaka lociranih po obodu kružnice [22]

Za jednostavnije Hough transformacije radijus je poznat, te je njegova vrijednost za dani primjer jednaka $r = 1$. Izvršava se transformacija parametara stvarne sliku u transformirani koordinatni sustav. Transformacija će biti izvršena koristeći matematičke izraze prema (4.46), (4.47), (4.48) i (4.49):

$$(x - a)^2 + (y - b)^2 = r^2 \tag{4.46}$$

$$x^2 - 2xa + a^2 + y^2 - 2yb + b^2 = r^2 \tag{4.47}$$

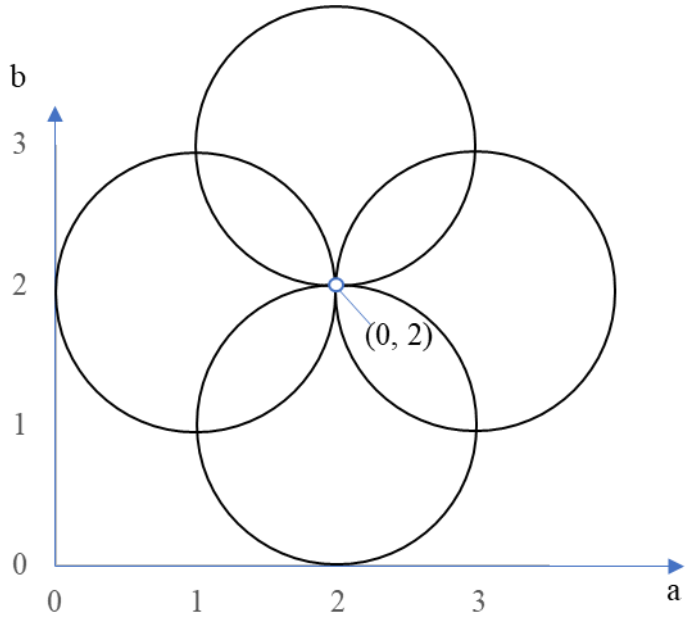
$$a^2 - 2xa + x^2 + b^2 - 2yb + y^2 = r^2 \tag{4.48}$$

$$(a - x)^2 + (b - y)^2 = r^2 \tag{4.49}$$

Obzirom na poznatu vrijednost r preko izraza (4.49) dobiva se izraz (4.50) koji se koristi za oderađivanje u transformiranom prostoru.

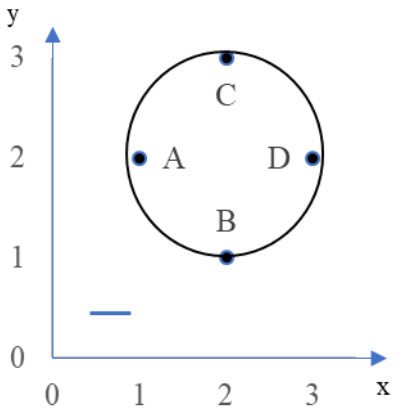
$$(a - x)^2 + (b - y)^2 = 1 \tag{4.50}$$

Iz točaka u novom prostoru parametri *x* i **y** su varijabilni. Primjenom Houghove kružne transformacije prikazane na slici 4.20 dana je točka visoke gustoće koja predstavlja centar traženog kruga poznatog radijusa r [22].

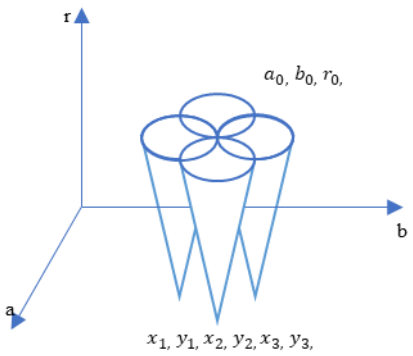


Slika 4.20 Houghova kružna transformacija za detekciju centra traženog kruga [22]

Konačno, rezultat je prikazan na slikama 4.21 i 4.22. Točke koje su korištene u akumulatoru naznačene su na obodu kruga.



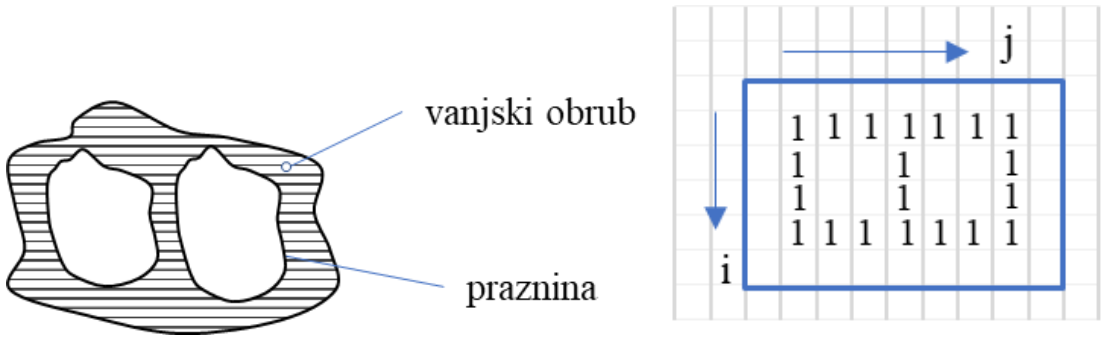
Slika 4.21 Detektirani krug sa pripadnim točkama [22]



Slika 4.22 Hough transformacija za krugove kod kojih nije poznata vrijednost radijusa [22]

4.3. Suzukijev algoritam za praćenje konture

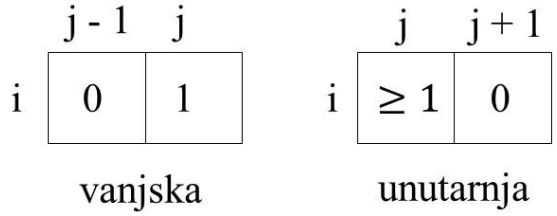
Godine 1985. zahvaljujući Satoshi Suzukiju koji publicira rad na temu "Topološka struktura analize binarnih digitalnih slika praćenjem konture" (engl. *Topological structural analysis of digitized binary images by border following*) predstavljena su dva algoritma za izdvajanje konture. Upravo Suzukijev algoritam predstavlja temelj za detekciju kontura, koja se koriste u bibliotekama otvorenog koda, primjerice Open CV. Suzukijev algoritam jedan je od prvih algoritama koji definira hijerarhijsku vezu između granica [23]. Prije samog objašnjenja algoritma potrebno je razumjeti osnovne parametre. Slika 4.23 prezentira binarnu sliku koju algoritam koristi kao ulazni parametar pri određivanju konture.



Slika 4.23 Binarni zapis granice objekta [23]

Funkcija $f(i,j)$ se sastoji od vrijednosti piksela na lokacijama (i,j). Pritom se svakoj pronađenoj granici na putu dodjeljuje jedinstven broj kojeg koji se rješava primjenom NBD (negativne binomne distribucije). Na mjestu gdje se nalazi granica pretpostavljena vrijednost distribucije odgovara vrijednosti 1. Ostatak granice je numeriran sekvencijalno. Skeniranje slike izvršava

se s lijeva na desno pri čemu se detektiranim pikselima dodjeljuju vrijednosti prema slici 4.24 [20]. Pritom se odlučuje je li pronađeni piksel pripada vanjskom rubu ili granici praznine.



Slika 4.24 Određivanje vanjske i unutarnje granice na temelju Suzukijeva algoritma [23]

Postupak daljnje analize može biti opisan kroz 4 koraka koja su grafički prezentirana na slici 4.25.

1. *Korak :*

   Ukoliko pronađeš vrijednosti $f_{i,j} = 1$ *i* $f_{i,j-1} = 0$ *dodjeli lokaciji* (i_2, j_2) *vrijednosti* $f_{i,j-1}$

   Inače je pronađena unutarnja kontura

   Postavi vrijednosti (i_2, j_2) *na vrijednosti* $f_{i,j+1}$ *te postavi* $f_{i,j}$ *kao roditelja ako vrijedi* $f_{i,j} > 1$

   Inače prijeđi na korak 3

2. *Korak :*

   2.1. *Počni od* (i_2, j_2) *i posjeti sva susjedstva od* (i, j) *u smjeru kazaljke na satu. Ako nađeš vrijednost ≠ 0 i notiraj ih kao* (i_1, j_1).

   Inače postavi $f_{i,j} = -$ *NBD i idi na korak 4.*

   2.2. *Postavi* $(i_2, j_2) = (i_1, j_1)$ *i* $(i_3, j_3) = (i, j)$

   2.3. *Kreni gledati sljedeći element piksela* (i_2, j_2) *u smjeru obrnutom smjeru kazaljke na satu. Ponovno prijeđi u susjedni element* (i_3,j_3) *i obiđi susjedstva u smjeru suprotnom od smjera kazaljke na satu kako bi našao prvu vrijednost različitu od 0 i postavi je na* (i_4, j_4)

   2.4. *Promjeni vrijednost trenutnog piksela* (i_3, j_3) *prema sljedećem:*

2.4.1. Ako je vrijednost piksela (i_3, j_{3+1}) jednaka 0, ovaj piksel pripada vanjskom području te njegovu vrijednost postavi na $-NBD$

2.4.2. Ako je vrijednost piksela $(i_3, j_{3+1}) \neq 0$, postavi trenutni piksel na vrijednost NBD

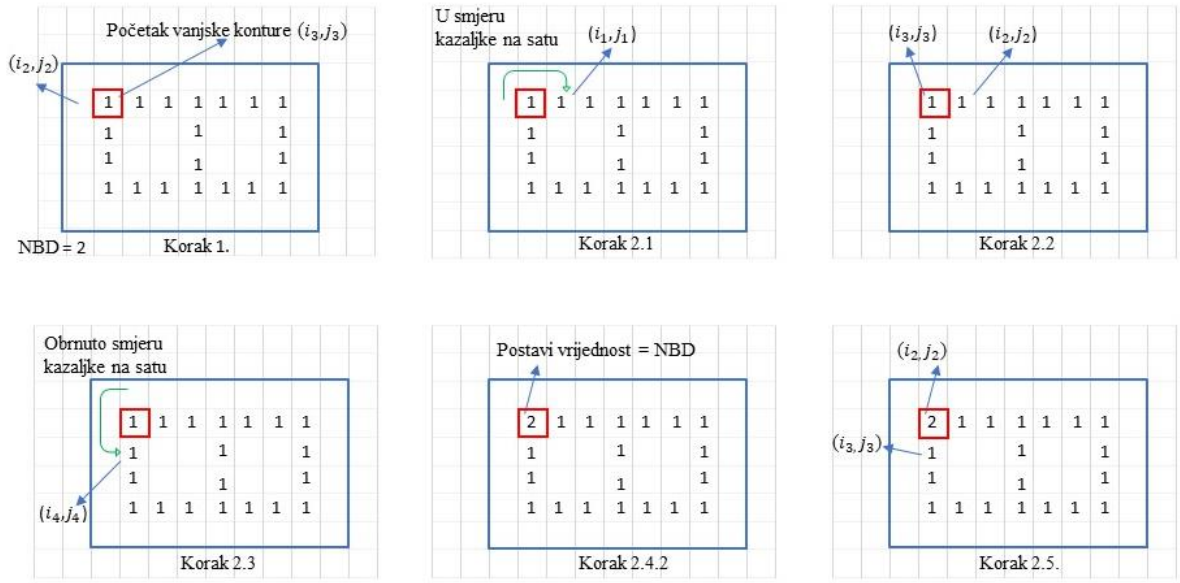
Inače ne mijenjaj vrijednost piksela

2.5 Ako se u koraku 2.3 ponovno vratiš u početnu točku odnosno ako vrijedi $(i_4, j_4) = (i, j)$ i $(i_3, j_3) = (i_1, j_1)$ idi na korak 3.

Inače postavi $(i_2, j_2) = (i_3, j_3)$ i $(i_3, j_3) = (i_4, j_4)$ i idi natrag na korak 2.3

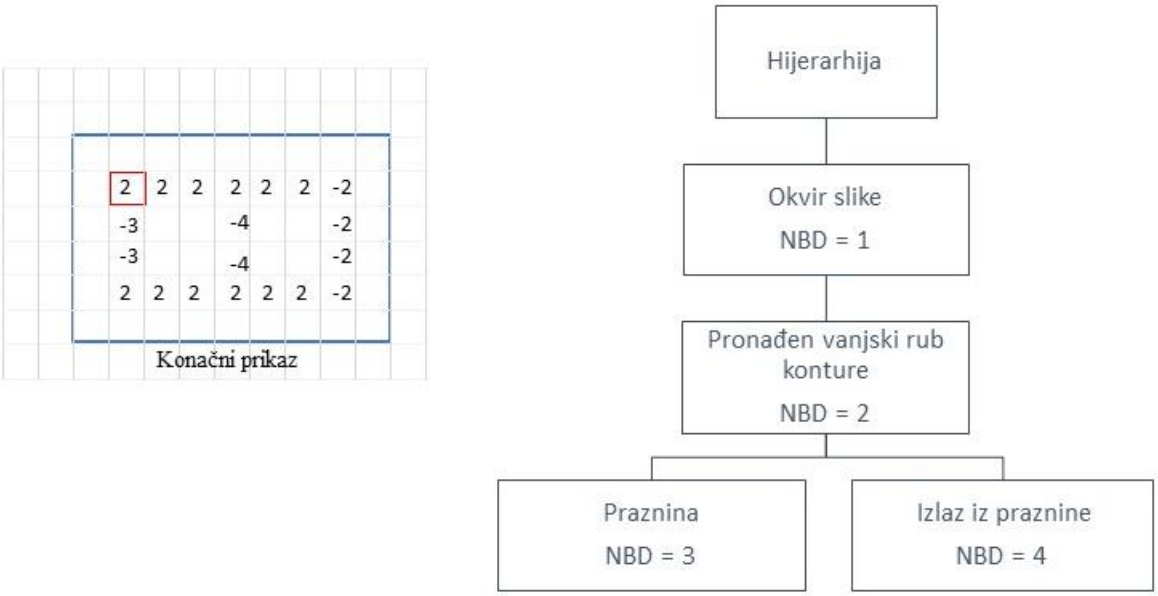
3. *Korak :*

Ako je $f_{i,j}! = 1$ postavi $| f_{i,j}| = roditelj$ te počni skenirati od sljedećeg piksela (i, j+1). Kad dosegneš donji desni kut označi kraj.



Slika 4.25 Faze detekcije konture primjenom Suzukijevog algoritma

Ponavljanjem predstavljenih koraka dobiva se sljedeći izlaz. Na slici 4.26 nalazi se hijerarhijski prikaz detekcije konture iz primjera sa slike 4.23



Slika 4.26 Konačni rezultat detekcije konture primjenom Suzukijevog algoritma

5. ODABIR I IMPLEMENTACIJA VIZIJSKOG SUSTAVA

Cilj ovog rada je predstavljanje algoritama koji bi imali mogućnost robusnog prepoznavanja i kontrole ispitnih uzoraka na nesređenim dvodimenzionalnim scenama. Zadatak je potrebno riješiti primjenom biblioteka otvorenog koda (engl. *Open Source*). Za rješenje danog zadataka korišten je programski jezik Python.

5.1. Odabir biblioteke za razvoj algoritma

U okviru ovog rada algoritmi otvorenog koda bit će primijenjeni za učitavanje, digitalnu obradu i analizu slike s ciljem detekcije i identifikacije željenih objekata.

Pri odabiru biblioteke za razvoj algoritma, morali su biti zadovoljeni sljedeći kriteriji:

- Točnost i pouzdanost konačno razvijenog rješenja u smislu vremenske potrošnje te učestalosti održavanja.
- Odabrana biblioteka mora biti široko primijenjena te općeprihvaćena u području računalnog vida.
- Biblioteka mora biti aktivno upotrebljavana, imati planiran razvoj te razvijenu podršku.
- Programski jezik.

Razvoj vizijskog sustava koji koristi besplatne biblioteke otvorenog koda ne implicira da je ovaj postupak besplatan niti jednostavan stoga je prije početka obrade podataka potrebno obratiti pažnju na odabir odgovarajuće biblioteke. Ako se u obzir uzmu biblioteke koje se razvijaju dulji vremenski period , izvjesno je da se kod ovih biblioteka sužava prostor za greške. To ne negira činjenicu kako je svaku biblioteku potrebno gledati sa spoznajom kako greške postoje, no ukoliko postoji podatak kako je održavanje željene biblioteke napravljeno dovoljno često, prostor za greške se smanjuje, što biblioteci daje dodatnu pouzdanost. Iako postoje mnogi izvori, primjerice forumi te otvoreni kod, biblioteke otvorenog koda nemaju službenu tehničku podršku, stoga ne možemo računati na pomoć u slučaju zastoja. Nadalje, važno je kako odabrana biblioteka iza sebe već ima određen broj obrađenih projekata slične problematike čime se dobiva kvalitetna povratna informacija o primijenjenosti i postignutoj kvaliteti tijekom primjene. Zadnja naznačena, međutim ne manje važna, je primjena odgovarajućeg te općeprihvatljivog programskog jezika.

Odabrana biblioteka u kojoj je razvijena algoritamska metoda za rješavanje danog zadatka je OpenCV.

Razvijen 1999. godine od američke tvrtke Intel, OpenCV (engl. *Open Source Computer Vision Libary*) predstavlja najpopularniju biblioteku otvorenog koda koja se koristi kod računalnog vida i strojnog učenja te sadrži više od 2,500 algoritama specijaliziranih za računalni vid i strojno učenje. U kolovozu 2012. podršku za OpenCV preuzela je neprofitna zaklada OpenCV. Unutar dane biblioteke izdana je brojna literatura za lakše snalaženje, a sve su funkcije jako dobro dokumentirane. OpenCV podržava učitavanje slika izravno iz kamere, mrežnim putem ili učitavanjem datoteka za pohranu u gotovo svim formatima. Koristi algoritme za pristup značajkama, detektiranje, filtriranje te raspoznavanje uzoraka. OpenCV je kompatibilan sa više programskih jezika i drugih razvojnih alata. Ova biblioteka ima BSD licencu odnosno besplatna je za akademske i komercijalne svrhe. Podržava operacijske sustave Windows, OS X, Linux, iOS, Android i Blackberry. U upotrebi je već 20 godina [24].

U okviru ovog rada u OpenCV su korišteni algoritmi za sljedeće aktivnosti:

- Učitavanje slike.
- Filtriranje.
- Detekcija ruba.
- Lokalizacija i detekcija oblika.
- Izdvajanje značajki detektiranog objekta.

Korištena verzija biblioteke je Python 3.8.

5.2. Odabir vizijskog sustava za testiranje ispitnih uzoraka

Vizijski sustav za analizu uzoraka koristi fiksno postavljenu kameru smještenu unutar zatvorene komore. Akvizicija slike postiže se korištenjem industrijske kamere koja je na računalo spojena preko USB sučelja. Pohrana snimljenih scena izvršavala se primjenom programa IC Capture. Segmentacija slike, morfološka transformacija, izdvajanje rubova te značajki kao što je broj zubi na obodu riješeni su korištenjem algoritma razvijenog na računalu.

Odabrana industrijska monokromatska kamera proizvod je tvrtke *The Imaging Source Europe Gmbh.* Korišten je model DMK23UX174 prikazan na slici 5.1.

Slika 5.1 Industrijska monokromatska kamera DMK 23UX174 [25]

Neke od značajki odabrane kamere prikazane su tablicom 5.1.

Tablica 5.1 Tehničke karakteristike industrijske kamere DMK 23UX174 [25]

OPĆE ZNAČAJKE	
Osjetljivost	0,05 lx
Dinamički raspon	8/12 bit
Format slike (max)	1920×1200
OPTIČKE ZNAČAJKE	
Tip senzora	CMOS Pregius
Zatvarač	globalni
Veličina piksela	$h = 5{,}86\mu\text{m}, v = 5{,}86\mu\text{m}$
ELEKTRONIČKE ZNAČAJKE	
Sučenje	USB 3.0
Napon napajanja	4,5 VDC to 5,5 VDC
Trenutna potrošnja	prosječno 250 mA do 5 VDC
Automatska kontrola šarenice	×
Okidač	✓
I/Os	✓
MEHANIČKE ZNAČAJKE	
Dimenzije	H: 29 mm, W: 29 mm, L: 43 mm
Masa	65 g
PRILAGODBA	
Zatvarač	1/(100 000)s – 30 s

Objektiv korišten na kameri je američke tvrtke FUJINON te pripada CF-HA seriji koju u prvom redu odlikuju visoka optička rezolucija te razmjerno mali distorzijski faktor. Korišten je model CF12.5HA – 1 prikazan na slici 5.2 [26].

Slika 5.2 CF12.5HA-1 objektiv tvrtke FUJINON [26]

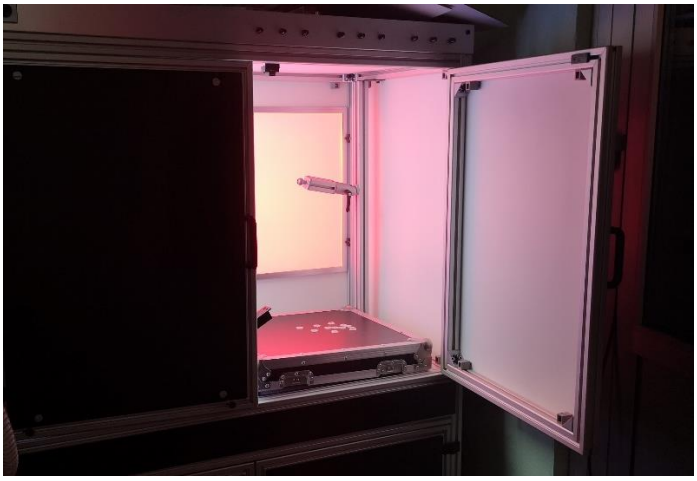
Za ovaj objektiv je poznato kako postiže dobre rezultate u akviziciji slika na malim udaljenostima. Ima nisku vrijednost f-stop broja koji predstavlja kvantitativnu mjeru odnosa žarišne daljine objektiva i otvora blende. Glavne značajke korištenog objektiva su prezentirane u tablici 5.2 [26].

Tablica 5.2 Tehničke specifikacije objektiva CF12.5HA-1 [26]

Fokalna duljina, mm	12,5
Promjer blende, f broj	F1,4 – F22
Kut pogleda	53,1°×41,2° (1" ili 2,54 cm)
Radnja udaljenost[2], mm	$\infty - 100$
Fokusiranje	Ručno
Navoj filtra, mm	M49 × 0,75
Veličina senzora (max)[4]	1"~2,54 cm (9,1μm)
Dimenzije, mm	Φ51× 68,5

Prikupljanje uzoraka slika je obavljeno na ispitnoj komori (Slika 5.3) pri kontroliranim uvjetima osvjetljenja. Komora se sastoji od više različitih izvora svjetlosti, koji se mogu međusobno kombinirati. Odabir izvora svjetla dobiven je iskustveno pri čemu je testirana kvaliteta

prikupljenih slika u ovisnosti o osvjetljenju. Korišteno je LED osvjetljenje s crvenog, plavog i zelenog spektra, te kombinacija navedenih. Izvor osvjetljenja je postavljen na različitim lokacijama (gore, dolje i bočno).

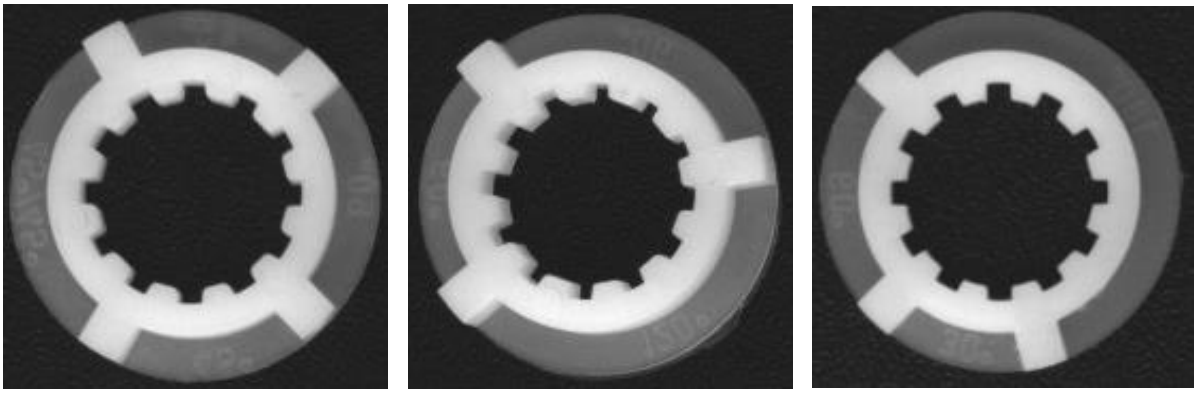


Slika 5.3 Ispitna komora

5.3. Odabir algoritma

Geometrijske značajke proizvoda predstavljaju jedinstven faktor za raspoznavanje. Prije prezentacije algoritma važno je predstaviti zadatak vezan za proces raspoznavanja testiranih uzoraka. Radi se o uloščima grebenaste sklopke. Uzorci su kružnog oblika nalik zupčaniku te približno bijele boje. Svaki uzorak ima jednak unutarnji provrt na kojem je raspoređen jednak broj zuba. U danom primjeru od značaja će biti raspored zuba na vanjskom obodu uloška. Na slici 5.4 prikazana su tri moguća tipa ispitnih uzoraka.

1. TIP **2. TIP** **3. TIP**



Slika 5.4 Ispitivani uzorci

Broj i raspored zubi određuje izvedbu sklopke, odnosno gotovog proizvoda. Važno je uočiti kako su vanjski zubi postavljeni na tankoj stijenci uzorka, koji ima manji stupanj intenziteta svjetline piksela. U postupku pretprocesiranja, tanku stijenku je bilo potrebno ukloniti s ciljem ispravne detekcije zubi. Ovaj korak je posebice važan u slučaju kada bi došlo do preklapanja uzoraka, čime bi intenzitet piksela na dijelu preklapanja bio znatno viši. Eliminacijom tanke stijenke možemo umanjiti osjetljivost na preklapanje. Ostale dimenzijske veličine različitih uložaka prikazane su u tablici 5.3.

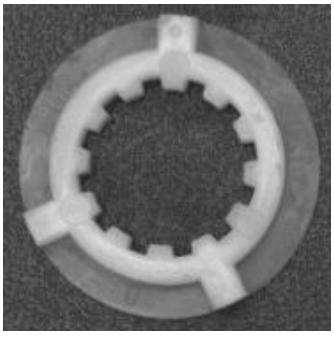
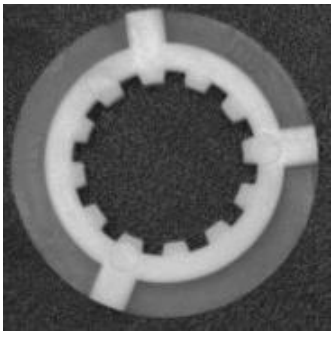
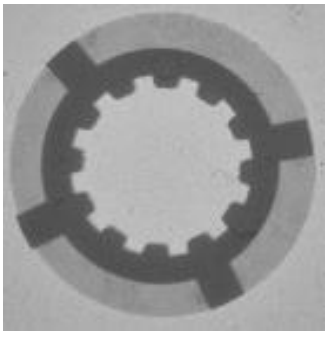
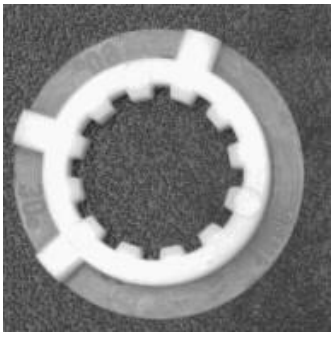
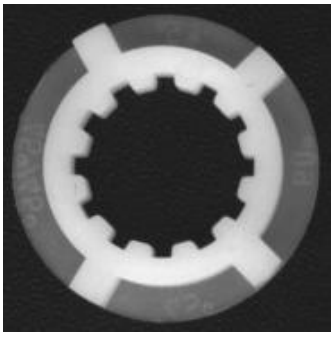
Tablica 5.3 Prikaz glavnih parametarskih veličina za identifikaciju

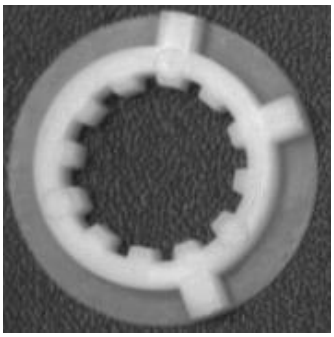
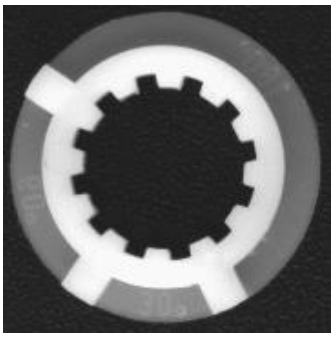
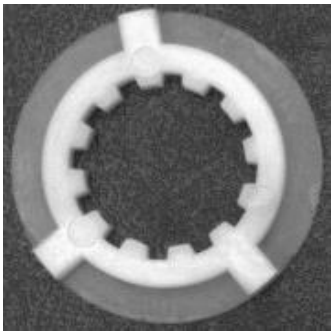
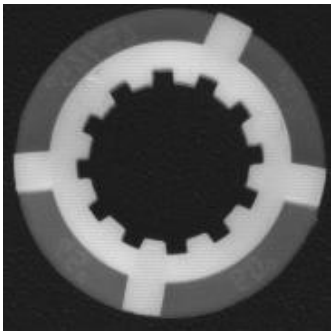
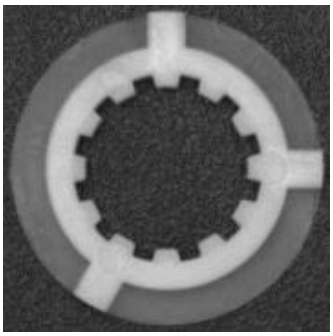
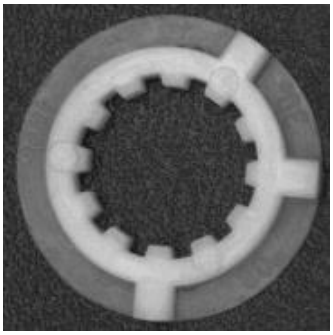
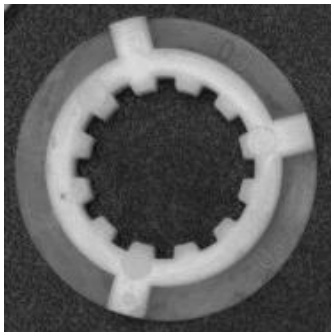
	1.TIP	2.TIP	3.TIP
Broj zubi	4	3	3
Kutni raspored zubi, °	90, 75, 120, 75	120, 90, 150	210, 90, 60

5.3.1. Osvjetljenje

Unutar komore koja nudi kontrolirane uvjete osvjetljenja iz više izvora snimljeni su testni uzorci prezentirani u tablici 5.4.

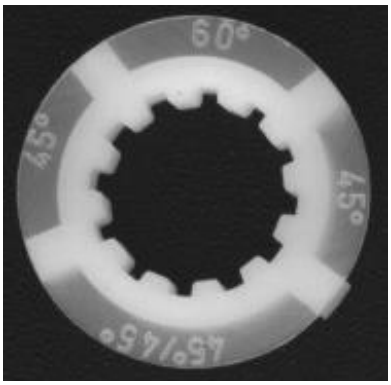
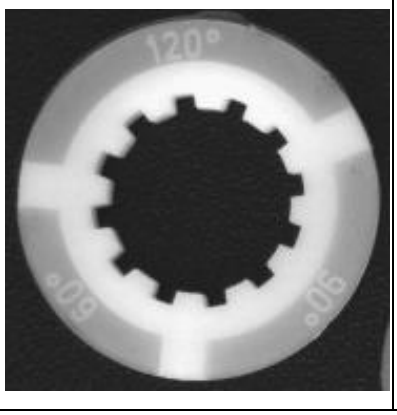
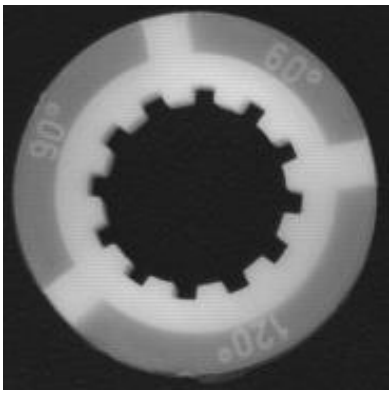
Tablica 5.4 Određivanje uvjeta osvjetljenja

Vrsta svjetla	Osvjetljenje odozgo	Bočno osvjetljenje	Osvjetljenje odozdo
Bijelo svjetlo			
Crveno svjetlo			

Plavo svjetlo	 	
Zeleno svjetlo	 	
Bijelo + Crveno		
Bijelo + Plavo		
Bijelo + Zeleno		

Postupak određivanja ispravnog osvjetljenja je jednostavan te vremenski kratak. Bez obzira na trivijalnost postupka on se ne smije zanemariti. Njegova važnost ogleda se u osiguravanju ulaznih podataka koji se koriste pri obradi i analizi slikovnih podataka algoritmima koji mogu biti manje računski intenzivni.

Za zadani objekt najpovoljnije se pokazalo bočno osvjetljenje kod kojeg smo dobili slike na kojima je objekt izražen a pozadina gotovo crne boje. Primjenom gornjeg osvjetljenja dobivene su slike kod kojih je izražena refleksija svijetla o površinu pa imamo veću količinu šuma. Tijekom snimanja cilj je predmet potpuno odvojiti od pozadine, ukoliko je to moguće, kako bi se formirali uniformno osvjetljenje za izdvajanje značajki objekta bez utjecaja pozadine. Primjenom osvjetljenja odozdo intenzitet uzorka je dosta blizak iznosu intenziteta pozadine zbog čega je i ova metoda odbačena. Osvjetljenje koje je dalje testirano je bočno osvjetljenje crvenog, plavog i zelenog spektra obzirom da su dali najbolje rezultate tijekom prvog ispitivanja. Drugo ispitivanje je provedeno na okrenutim uzorcima. Slika 5.5 prikazuje prikupljene rezultate eksperimenta.

Crveno bočno osvjetljenje	Plavo bočno osvjetljenje	Zeleno bočno osvjetljenje
		

Slika 5.5 Prikaz ispitnih uzoraka snimljenih bočnim osvjetljenjem različite boje

Kod ovog eksperimenta ispitivao se utjecaj osvjetljenja na granice zubi smještenih na obodu te stjenke na obodu na kojoj je naznačen raspored zubi. Korišteno uniformno osvjetljenje crvene boje minimizira utjecaj obodne stijenke na rub zuba. Zbog navedenih karakteristika za daljnju analizu zadanog objekta odabire se crveno bočno osvjetljenje.

5.3.2. Predložene algoritamske metode

U okviru ovog diplomskog rada napravljen je razvoj dvije algoritamske metode koje su bile prikladne za rješavanje danog problema.

Prva promatrana karakteristika koja se uzela u obzir je svakako boja te intenzitet piksela na lokacijama zubi u odnosu na okolinu. Druga karakteristika koja se uzela u razmatranje je udaljenost zuba u odnosu na ostatak konture. Pritom se prvenstveno misli na dio uloška kojem ne pripada tankoj plastičnoj stjenci . Obzirom na poznate podatke izrađena su dva algoritma koja se temelje na različitim metodama:

- Algoritam temeljen na generaliziranim Houghovim transformacijama i analizi intenziteta obodnih piksela.
- Algoritam temeljen na pronalasku konture.

Temeljni princip rada te matematička notacija oba algoritama obrađeni su prethodnim poglavljima. Algoritmi su razvijeni u programskom jeziku Python te testirani u PyCharm integriranom razvojnom okruženju ili kraće IDE (engl. *Integrated Development Environment*) PyCharm. U nastavku rada će biti obrađen razvoj koda predloženih algoritama i prezentirati dobiveni rezultati.

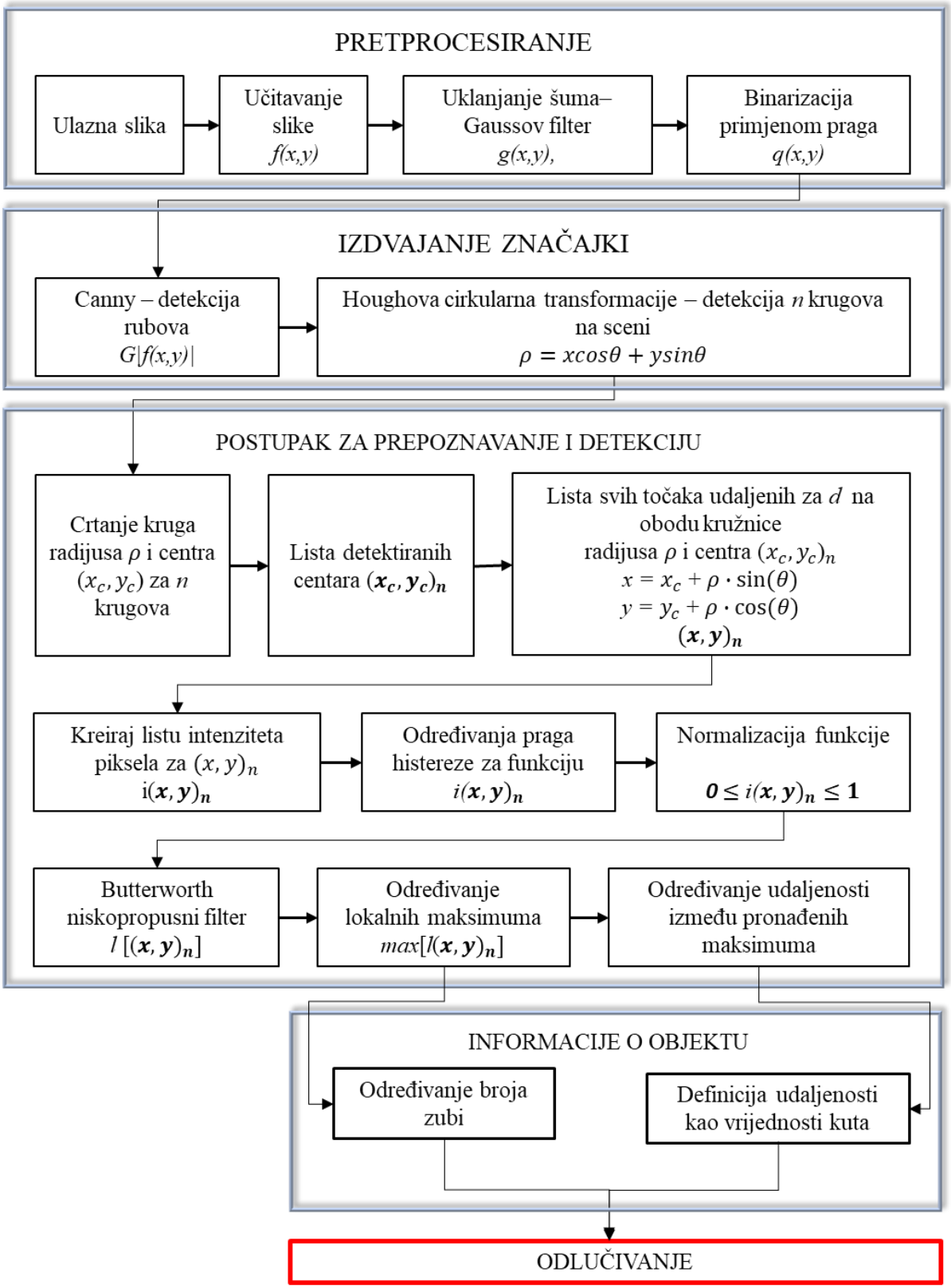
5.4. Algoritam temeljen na generaliziranoj Houghovoj transformaciji i analizi intenziteta obodnih piksela

Odabrani postupak moguće je opisati u nekoliko glavnih dijelova. Objašnjeni su temeljni dijelovi algoritma te pregled različitih faza obrade. Predložena metoda prepoznavanja izdvojenih značajki temelji se na analizi intenziteta obodnih piksela. U zadnjem dijelu poglavlja dani su rezultati predloženog razvijenog algoritma.

5.4.1. Dijagram toka algoritma

Rad algoritma temeljenog na Houghovoj transformaciji te analizi piksela prikazan je na slici 5.20. Postupak se može opisati u nekoliko glavnih dijelova:

1. Pretprocesiranje.
2. Izdvajanje značajki.
3. Postupak za prepoznavanje i detekciju.
4. Odlučivanje.



Slika 5.6 Dijagram operacija algoritma temeljenog na Houghovoj transformaciji i analizi piksela

5.4.2. Prikaz rada algoritma kroz glavne faze

Prvi dio algoritma bavi se postupcima segmentacije te izdvajanja značajki. Primjenjuje se Gaussov filtar, postavljanje praga te detekcija ruba primjenom Canny operatora. Rezultati su prezentirani na slici 5.7, a dobiveni su primjenom koda prikazanog na slici 5.8. Prikazane faze neophodne su prije daljnje analize temeljene na generaliziranoj Houghovoj transformaciji.

Slika 5.7 Faza segmentacije i izdvajanja značajki

```
import cv2
import numpy

img = cv2.imread('uzorak.jpg',0)
blur = cv2.GaussianBlur(img, (5, 5), cv2.BORDER_DEFAULT)
ret, threshold = cv2.threshold(blur, 132, 255, cv2.THRESH_BINARY)
edges = cv2.Canny(threshold, 100, 255)

cv2.imshow("Pretprocesiranje", numpy.hstack(img, blur, threshold, edges))
cv2.waitKey(0)
cv2.destroyAllWindows()
```

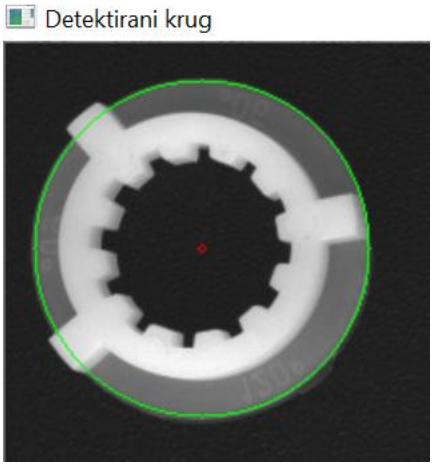
Slika 5.8 Prikaz koda korištenog pri detekciji ruba ispitivanog objekta

Primjenom Houghove kružne transformacije detektiraju se prisutni krugovi. Detekcija kruga izvršava se pozivom funkcije cv.HoughCircles koja je definirana sljedećim argumentima:

- *image* - ulaznom pretprocesiranom monokromatskom slikom.
- *method* - definira metodu detekcije - cv.HOUGH_GRADIENT
- *dp* - dp = 1 rezolucija akumulatora
- *min_dist* - minimalna udaljenosti između centara detektiranih krugova
- *param_1* - gornja vrijednost praga
- *param_2* - vrijednost praga za određivanje centra
- *min_radius* - najmanji mogući detektirani radijus

- *max_radijus* - maksimalni mogući detektirani radijus [27].

Detektirani krug prikazan je na slici 5.9. a pripadni kod koji se koristio pri njegovoj detekciji prikazan je na slici 5.10.

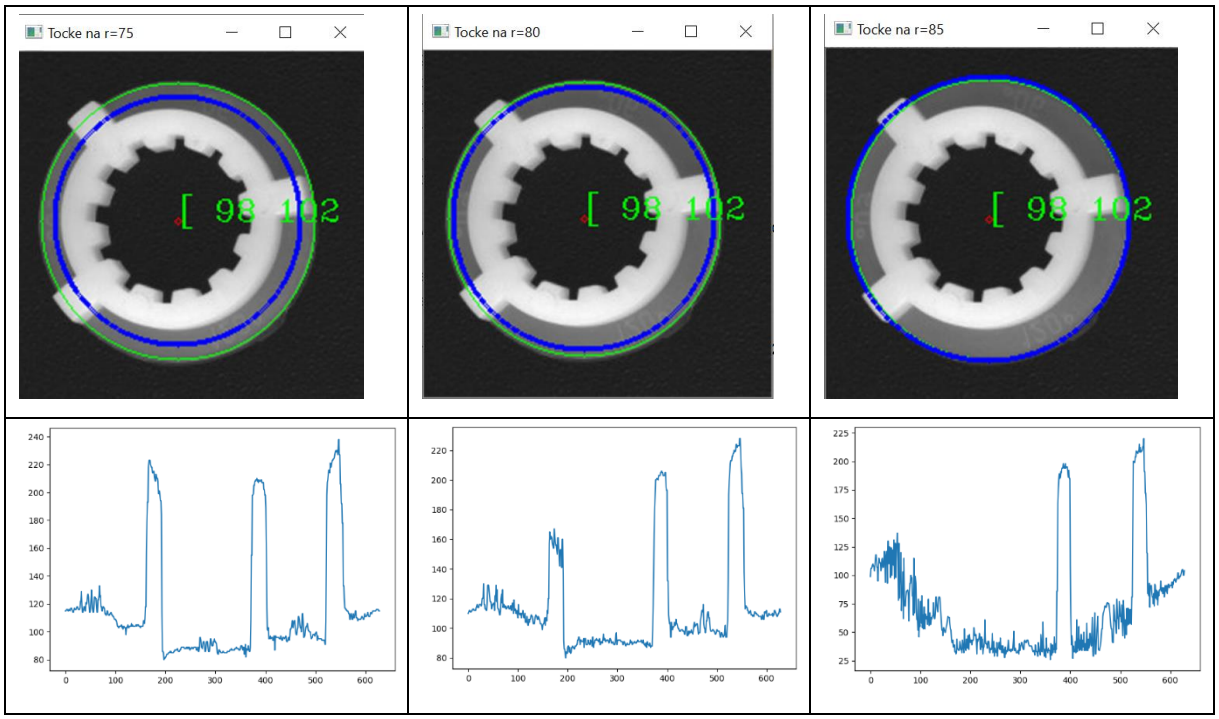


Slika 5.9 Detekcija kruga primjenom Houghove transformacije

```
circles = cv2.HoughCircles(img,cv2.HOUGH_GRADIENT,1,100,
param1=50,param2=60,minRadius=70,maxRadius=100)
circles = np.uint16(np.around(circles))
for i in circles[0,:]:
    # nacrtaj krug radijusa jednakog radijusu detektiranog kruga
    cv2.circle(cimg,(i[0],i[1]),i[2],(0,255,0),1)
    # nacrtaj radijus r=2 koju predstavlja centar
    cv2.circle(cimg,(i[0],i[1]),2,(0,0,255),1)
```

Slika 5.10 Detekcija krugova primjenom Houghove transformacije

Primjenom kružne Houghove transformacije dobiva se izlazna lista koordinata koji se sastoji od seta vrijednosti: $x_{c,}$ y_c i r za svaki detektirani krug. Pomoću dobivenih vrijednosti za detektiranje centara provelo se ispitivanje idealne veličine radijusa koji će dati najbolje izlazne vrijednosti razlike piksela. Rezultati s pripadnim korelacijskim grafovima su prikazani na slici 5.11.



Slika 5.11 Određivanje vrijednosti radijusa za daljnje provođenje testiranja

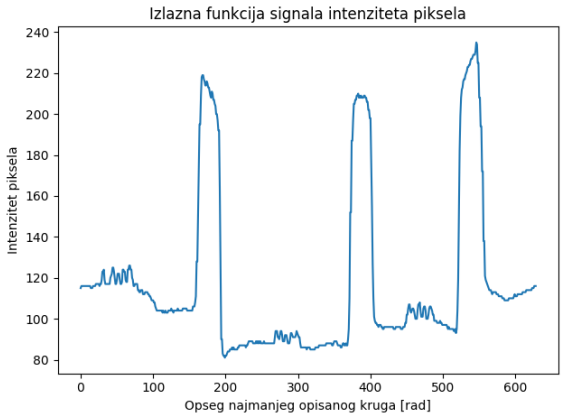
S obzirom na veličinu promatranih uzoraka te položaj kamere u prostoru odabran je radijus $r = 75$. Odabrani radijus na korelacijskom grafu pokazuje jasnu razdiobu točaka koje pripadaju, odnosno ne pripadaju zubu. Kod radijusa većeg iznosa uočava se proporcionalan rast iznosa bijele boje pozadinskih piksela koja kod uzoraka u nesređenoj sceni može stvarati problem. Već pri povećanju radijusa na vrijednost $r = 80$ kod detektiranog prvog zuba uočava se velik šum čiji intenzitet piksela raste. Daljnjim porastom radijusa došlo bi do ujednačenja intenziteta piksela zuba i pozadine, te zub ne bi bio detektiran.

Sljedeći korak je formiranje liste točaka koje se nalaze na zadanom radijusu $r = 75$ uz poznate centre očitane iz vektora liste koordinata središta. Kreirana je funkcija prema slici 5.12 koja obilazi sve točke na zadanom radijusu do pune revolucije kruga uz korak 0,01. Potrebno je naglasiti kako programski jezik Python koristi radijane pri definiranju kutova.

```
import math

lista_centar = []
lista_tocaka = []

for circle in circles[0]:
    _intenziteti = []

    xc = float(circle[0])
    yc = float(circle[1])
    cr = float(circle[2])

    radius = 75.0
    for a in np.arange(0,2*math.pi,0.01):
        x = xc + radius * math.sin(a)
        y = yc + radius * math.cos(a)
        pixelVal = img[int(y),int(x)]
        _intenziteti.append([a, pixelVal])
        cv2.circle(cimg, (int(x),int(y)), 1, (255,0 ,0),1)

    cv2.putText(cimg, str(circle), (circle[0],circle[1]),
cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (0,255,0))

    lista_tocaka.append(_intenziteti)
    lista_centar.append([xc,yc])
```

Slika 5.12 Formiranje liste točaka na obodu kružnice

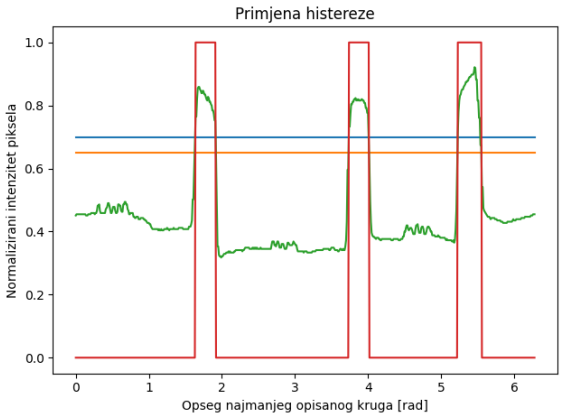
Intenzitet piksela u listi kreiranih točaka prikazan je na slici 5.13.



Slika 5.13 Intenzitet piksela u listi kreiranih točaka

Idući korak je normalizacija. Normalizacijom se ostvaruje skaliranje vrijednosti piksela na način da najintenzivnija vrijednost piksela poprimi vrijednost 1, a minimalna 0, čime se osigurava neosjetljivost na linearnu promjenu rasvjete. Nakon provedene normalizacije primjenjuje se histereza koja primjenjuje postavljanje dvostrukog praga. Dvostruki prag se

postavlja s ciljem ispravnog definiranja intenziteta obodnih piksela. Rezultati histereze prikazani su na slici 5.14.



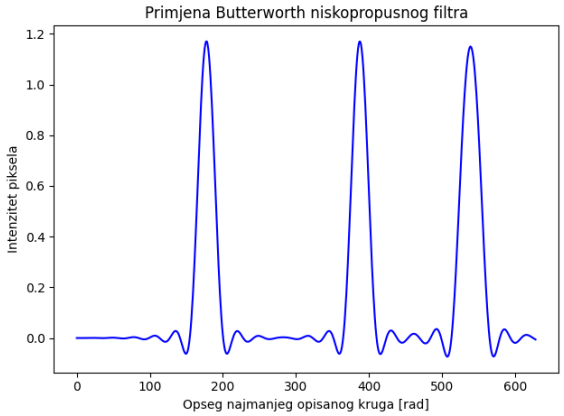
Slika 5.14 Primjena histereze kod detekcije rubova

Histereza korištenjem dvostrukog praga definira točnu lokaciju zuba analizom intenziteta obodnih piksela. Vrijednosti koje se nalaze iznad gornjeg praga predstavljaju točne lokacije zubi. Sve što se nalazi ispod donjeg praga pripada pozadini i sigurno nije zub. Vrijednosti između gornjeg i donjeg praga promatraju se zasebno. Ukoliko funkcija u području između gornjeg i donjeg praga raste, ove točke klasificiraju se kao vrh odnosno pozadina sve do trenutka prelaska gornjeg praga. Ukoliko funkcija u pojasu histereze pada, ovi intenziteti piksela klasificiraju se kao zub sve do trenutka prelaska donjeg praga. Na slici 5.15 prikazan je kod koji je korišten za postupak histereze. Ukoliko funkcija prijeđe gornji prag histereze poprima vrijednost jedinice. Vrijednost funkcije, odnosno signala, ostaje jednaka sve dok funkcije ne padne ispod vrijednosti donjeg praga histereze, nakon čega joj se dodjeljuje vrijednost nula.

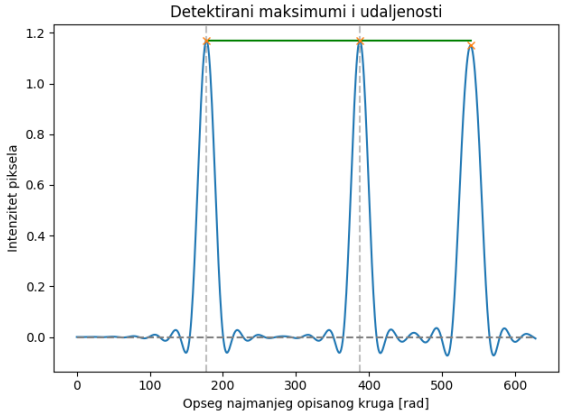
```
def ups_downs(y):

    try:
        summed = np.cumsum(y)
        d = np.sign(np.diff(summed))
        ud = np.flatnonzero(d)
        uds = d[ud]
        change = ud[np.r_[True, uds[1:] != uds[:-1]]]

        out = np.zeros(len(summed), dtype=int)

        out[1:][change] = d[change]
    except IndexError:
        out = y
    return out


def hyst(x, donji_prag, gornji_prag, initial = False):
    if donji_prag > gornji_prag:
        x = x[::-1]
        donji_prag, gornji_prag = gornji_prag, donji_prag
        rev = True
    else:
        rev = False
    hi = x >= gornji_prag
    lo_or_hi = (x <= donji_prag) | hi
    ind = np.nonzero(lo_or_hi)[0]
    if not ind.size:
        x_hyst = np.zeros_like(x, dtype=bool) | initial
    else:
        cnt = np.cumsum(lo_or_hi)
        x_hyst = np.where(cnt, hi[ind[cnt - 1]], initial)
    if rev:
        x_hyst = x_hyst[::-1]
    return x_hyst
```

Slika 5.15 Definiranje histereze

Središnje vrijednosti zubi koji su detektirani iz liste kreiranih točaka filtrirane su Butterworth niskopropusnim filtrom koji će dati točnu lokaciju lokalnog maksimuma zuba, odnosno središnje točke zuba. Prikaz intenziteta liste točaka nakon primjene filtra prikazan je na slici 5.16. Za detekciju lokalnih maksimuma korištena je funkcija *find_peaks*. Detektirani maksimumi izlazne funkcije te međusobne udaljenosti prikazane su na slici 5.17 te dobivene prema kodu prikazanom na slici 5.18.



Slika 5.16 Primjena Butterworth niskopropusnog filtra



Slika 5.17 Detektirani lokalni maksimumi i međusobne udaljenosti

```
from scipy.signal import find_peaks
from scipy.signal import butter, filtfilt

N = 1
Wn = 0.05
B, A = butter(N, Wn, output='ba')
smooth_data = filtfilt(B, A, h1[:1200])
plt.title('Primjena Butterworth niskopropusnog filtra')
plt.xlabel('Opseg najmanjeg opisanog kruga [rad]')
plt.ylabel('Intenzitet piksela')
plt.show()

x = np.array(smooth_data)
peaks, _ = find_peaks(x, 0.5)
y = [x[i] for i in peaks]

dist = [np.linalg.norm(np.array([peaks[i], y[i - 1]]) - np.array([peaks[i
- 1], y[i - 1]])) for i in
        np.arange(1, len(peaks))]
for i in range(len(dist)):
    plt.plot([peaks[i], peaks[i] + dist[i]], 2 * [y[i]], color='g')
    plt.axvline(peaks[i], linestyle='--', color='0.75')
plt.plot(x)
plt.plot(peaks, x[peaks], "x")
plt.plot(np.zeros_like(x), "--", color="gray")
plt.title('Detektirani maksimumi i udaljenosti')
plt.xlabel('Opseg najmanjeg opisanog kruga [rad]')
plt.ylabel('Intenzitet piksela')
plt.show()
```

Slika 5.18 Detekcija lokalnih maksimuma i određivanje udaljenosti

Broj detektiranih lokalnih maksimuma izravno daje broj zubi pronađenih na obodu uzorka. Udaljenost između detektiranih zubi normalizirana je na vrijednosti kuta. Konačni rezultati prikazani su na slici 5.19.

Slika 5.19 Rezultat prepoznavanja primjenom algoritma temeljenog na Houghovoj transformaciji

5.5. Algoritam temeljen na pronalasku konture

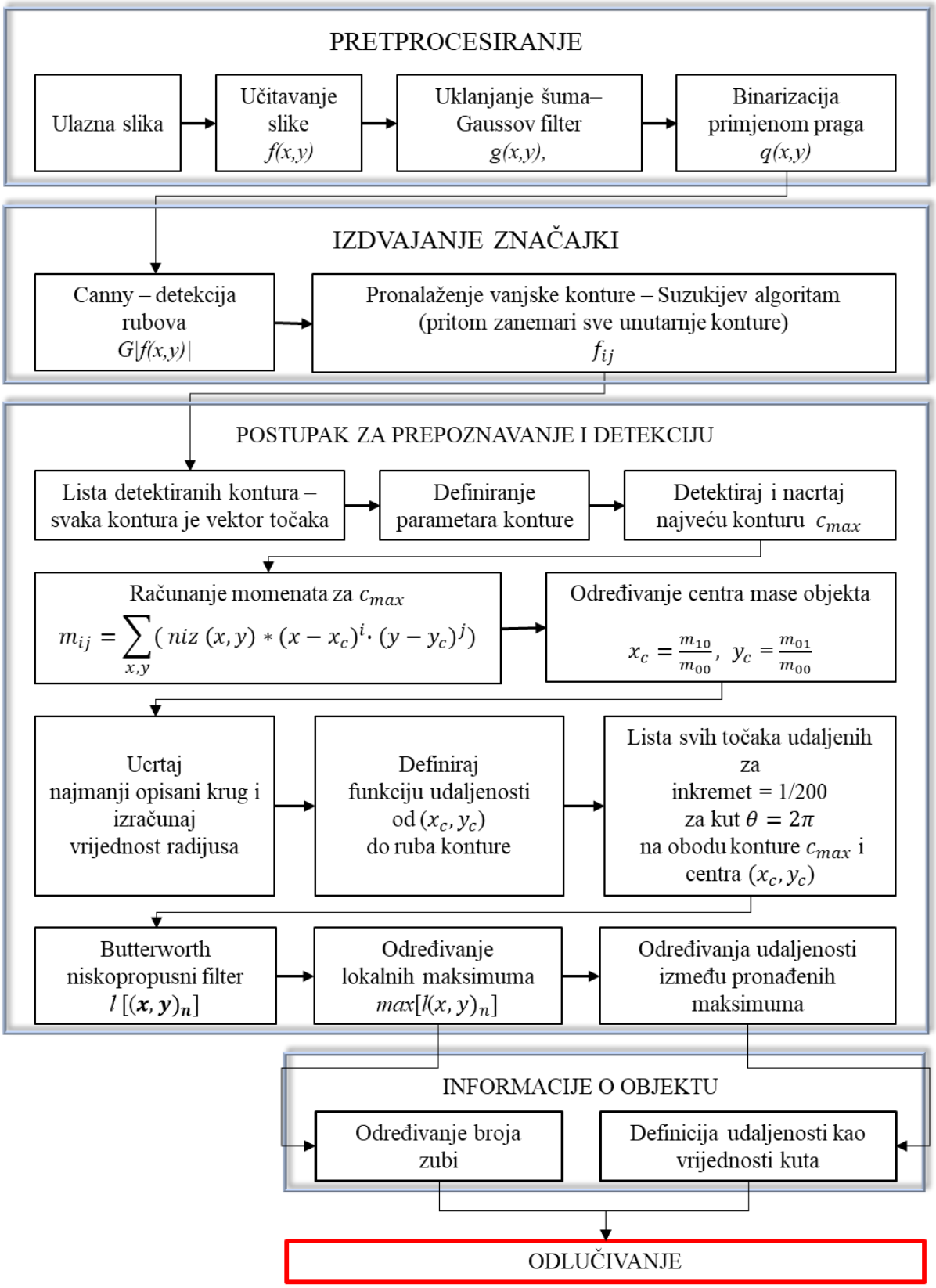
Kontura jednostavno može biti opisana kao krivulja opisana nizom sekanti čiji pikseli su međusobno povezani te imaju jednake intenziteta. Algoritam koristi konturu kao parametar identifikacije. Algoritam je računski intenzivniji u odnosu na prvi prezentirani algoritam koji izravno očitava vrijednosti piksela. Dijagram rada algoritma prikazan je na slici 5.20.

5.5.1. Dijagram toka algoritma

Rad algoritma temeljenog na pronalasku konture prikazan je na slici 5.20. Postupak provođenja algoritma je u pogledu glavnih faza izvođenja potpuno jednak. Sastoji se od postupaka:

1. Pretprocesiranje.
2. Izdvajanje značajki.
3. Postupak za prepoznavanje i detekciju.
4. Odlučivanje.

Razlike u razvijenim algoritamskim metodama nalaze se u načinu izdvajanja značajki te primjeni istih pri procesu odlučivanja.

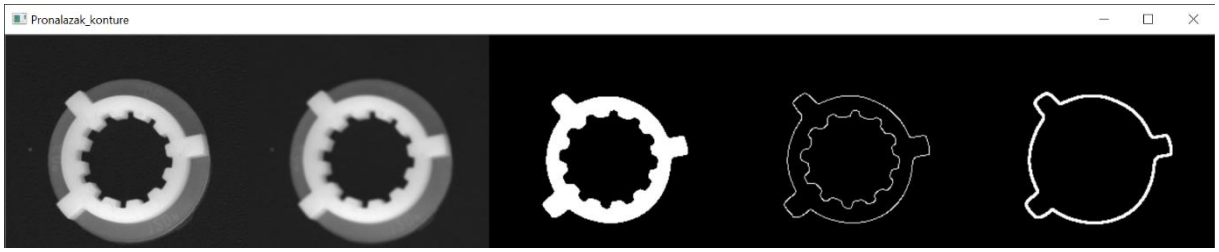


Slika 5.20 Dijagram operacija algoritma temeljenog na pronalasku konture

5.5.2. Prikaz rada algoritma kroz glavne faze

Faza pretprocesiranja sastoji se od obrade slike uklanjanjem šuma primjenom Gaussovog filtra dimenzija matrice 5×5, nakon čega je napravljena segmentacija slike primjenom praga koju slijedi postupak detekcije ruba primjenom Canny operatora. Važno je spomenuti kako je u biblioteci OpenCV pronalaženje konture definirano kao pronalazak bijelog objekta na crnoj podlozi što jasno ukazuje zašto su korištene spomenute faze pretprocesiranja.

Sljedeća faza algoritma bavi se izdvajanjem značajki pri čemu se kod zadanog problema misli na konturu. Algoritam koji se koristi zasniva se na Suzukijevom algoritmu, čiji princip rada je izložen u središnjem dijelu ovog rada. Funkcija u Pythonu koja izvršava radnju pronalaska konture prezentirana je na slici 5.21.



Slika 5.21 Faze obrade i detekcije vanjske konture

Faze obrade, detekcije te crtanja konture nalaze se u dijelu koda prikazanom na slici 5.22. Korištena funkcija cv.findContours() sastoji se od tri argumenta: ulazne slike koja je prethodno obrađena, načina detekcije konture te aproksimacij konture kao niza (x, y) koordinata koje kreiraju granicu objekta [29].

Funkcija cv.RETR EXTERNAL se koristi kako bi se detektirala samo vanjska kontura obzirom da nam u procesu prepoznavanja zadanog objekta unutarnja kontura nije od značaja. Primjenom cv.CHAIN APPROX SIMPLE uklonjene su nepotrebne detektirane točke konture.[29].

Parametri potrebni za detekciju objekta u izravnoj su vezi s vrijednostima značajki konture. Glavne značajke konture su:

- Momenti (engl. *Moments*):
  - Računanje prosječne težine intenziteta piksela na slici objekta primjenom funkcije cv.moments().
- Područje konture (engl. *Contour Area*):
  - Računanje područja unutar konture primjenom funkcije cv.contourArea.

- Duljinu konture (engl. *Contour Perimeter*):
  - Mjeri duljinu konture u pikselima primjenom funkcije cv.arcLength. Krivulja konture je aproksimirana nizom sekanti koje mogu biti izračunate iz koordinata poznatih točaka koje ih povezuju. Drugi argument u ovoj funkciji daje informaciju da li se radi o zatvorenoj konturi (True) ili samo krivulji (False).
- Aproksimacija konture (engl. *Contour Approximation*):
  - Izračunava se primjenom funkcije cv.approxPolyDP koja aproksimira oblik konture s manjim brojem vrhova. Drugi argument koji je naznačen kao epsilon predstavlja maksimalno odstupanje konture u odnosu na aproksimiranu konturu. Budući da predstavlja parametar točnosti, potrebno ga je pažljivo odabrati [28].

Određivanje glavnih značajki konture prikazano je kodom prema slici 5.22.

```
import numpy as np
import cv2 as cv

im = cv.imread('uzorak.jpg',0)
blur = cv.GaussianBlur(im, (5, 5), 0)
ret, thresh = cv.threshold(blur, 180 , 255, cv.THRESH_BINARY)
edges = cv.Canny(thresh, 100, 255)
contours, hierarchy = cv.findContours(edges, cv.RETR_EXTERNAL,
cv.CHAIN_APPROX_SIMPLE)

contour_list = []
for cnt in contours:
    epsilon = 0.015 * cv.arcLength(cnt, True)
    approx = cv.approxPolyDP(cnt, epsilon, True)
    area = cv.contourArea(cnt)
    if ((len(approx) > 5) & (len(approx) < 25) & (area > 50)):
        contour_list.append(cnt)
im1 = np.zeros_like(im)

c = max(contours, key=cv.contourArea)

M = cv.moments(c)
cX = int(M["m10"] / M["m00"])
cY = int(M["m01"] / M["m00"])

_, gear_radius = cv.minEnclosingCircle(c)
centroid = cX, cY
cv.drawContours(im1, contour_list, 0, (255, 255, 255), 2)
```

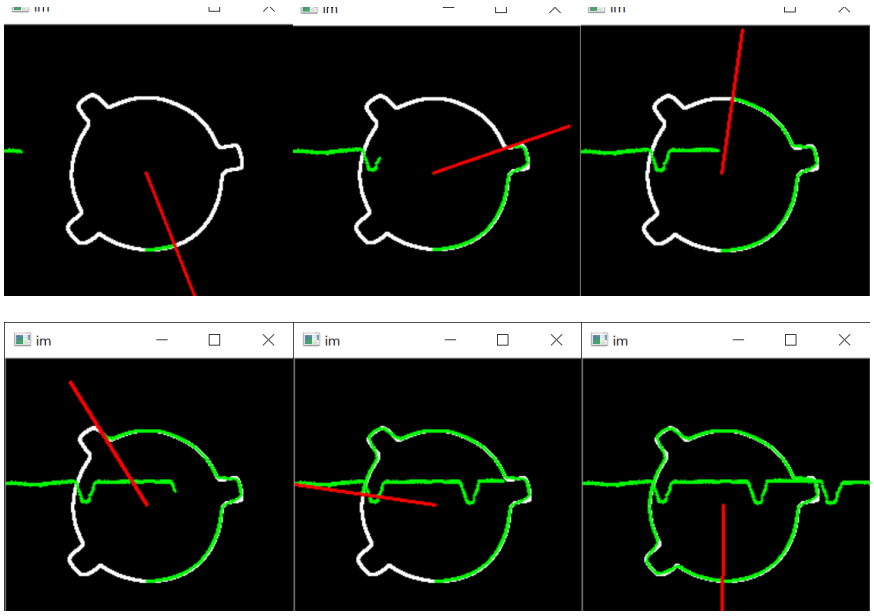
Slika 5.22 Prikaz pretprocesiranja te određivanja glavnih značajki konture

Mjerenje udaljenosti od centra dobivenog momentima do pronađenog ruba konture definirano je funkcijom prikazanom na slici 5.23.

```
def distance(a, b): return math.sqrt((a[0] - b[0]) ** 2 + (a[1] - b[1])
** 2)
```

Slika 5.23 Funkcija mjerenja udaljenosti od točke centra do ruba konture

Postupak kreiranja liste udaljenosti od centra prikazan je na slici 5.24. S ciljem promatranja izvođenja procesa izrađena je video animacija u kojoj se točno prepoznaju faze obilaska točaka konture do pune revolucije. Na slici je zelenom bojom prikazana ovisnost udaljenosti o vrijednosti kuta. Na mjestima gdje se nalazi zub jasno je vidljivo povećanje vrijednosti funkcije udaljenosti pomoću koje je moguće detektirati lokalne maksimume. Dio koda koji provodi mjerenje udaljenosti prikazan je na slici 5.25.



Slika 5.24 Proces kreiranja liste udaljenosti

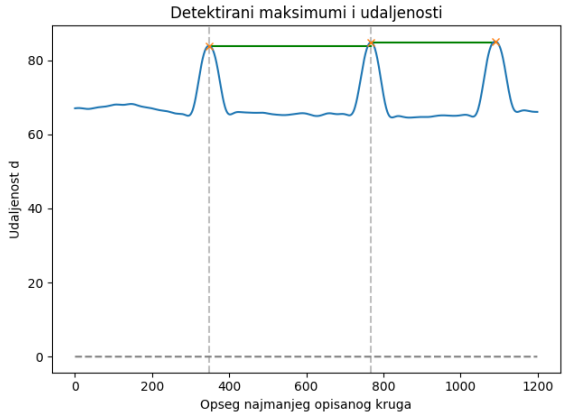
```
vid_writer = cv.VideoWriter('primjer', cv.VideoWriter_fourcc('M', 'J',
'P', 'G'), 60, (im.shape[1], im.shape[0]))
angle = 0
increment = 1 / 200
distances = []
display_im = im.copy()

while angle < 2 * math.pi:
        im_size = max(im.shape)
    ray_end = int(math.sin(angle) * im_size * 0.5 + cX),
int(math.cos(angle) * im_size * 0.5 + cY)
    center = cX, cY
    mask = np.zeros((im.shape[0], im.shape[1]), np.uint8)
    cv.line(mask, center, ray_end, 255, 5)
    gear_slice = cv.bitwise_and(im, im, mask=mask)
    ret, thresh = cv.threshold(cv.cvtColor(gear_slice,
cv.COLOR_BGR2GRAY), 0, 255, 0)
    contours, hierarchy = cv.findContours(thresh, cv.RETR_EXTERNAL,
cv.CHAIN_APPROX_SIMPLE)
    try:
        M = cv.moments(max(contours, key=cv.contourArea))
    except:
        print("Konture nisu detektirane ispravno. Potrebno je promjeniti
vrijednosti parametara")
        break
     edge_location = int(M["m10"] / M["m00"]), int(M["m01"] / M["m00"])
    cv.circle(display_im, edge_location, 0, (0, 255, 0), 2)
    edge_center_distance = distance(center, edge_location)
    graph_point = int(angle * 0.5 * im.shape[1] / math.pi),
int(edge_center_distance + 0.5 * gear_radius)
    cv.circle(display_im, graph_point, 0, (0, 255, 0), 2)
    distances.append(edge_center_distance)
    temp = display_im.copy()
    cv.line(temp, ray_end, (cX, cY), (0, 0, 255), 2)
    cv.imshow('im', temp)
    vid_writer.write(temp)
    k = cv.waitKey(1)
    if k == 27: break
    angle += increment
cv.destroyAllWindows()
```

Slika 5.25 Kreiranje liste udaljenosti od centra mase objekta do vanjske konture

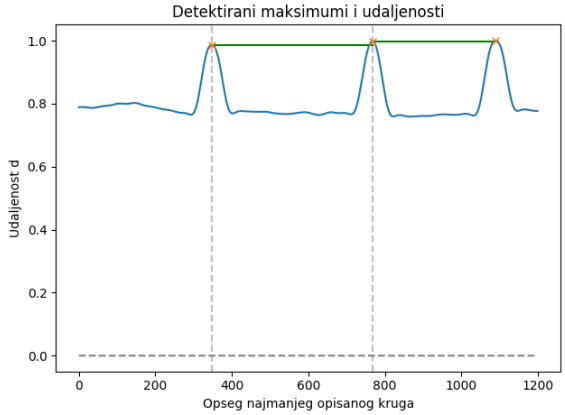
Ovisnost udaljenosti rubne konture grafički je prikazana na slici 5.26. Izlazni graf na apscisi ima iskazane vrijednosti kuta izražene u radijanima, a na ordinati ispitivane vrijednosti udaljenosti od centra. Izlazi signal je imao značajan šum te je detekcija broja vrhova bila otežana obzirom da se detektirao velik broj lokalnih maksimuma.

Za rješenje spomenutog problema korišten je Butterworth niskopropusni filter koji je zagladio funkciju. Broj lokalnih maksimuma direktno je određivao broj zuba, a udaljenosti među njima su normalizirane kao i vrijednosti kuta kao što je prikazano na slici 5.27.

Konačni rezultat prepoznavanja prikazan je na slici 5.28.



Slika 5.26 Detektirani lokalni maksimumi te njihova međusobna udaljenost primjenom algoritma temeljenog na pronalasku konture



Slika 5.27 Prikaz detektiranih zubi (označen sa x) te njihova međusobna udaljenost (zelena linija)

```
Lista kuteva je [120.0, 90.0, 150.0]
Broj zubaca =  3
Ima 3 zuba, sa udaljenostima 150, 120, 90

TOTAL TIME: 4.2124s

Process finished with exit code 0
```

Slika 5.28 Rezultat prepoznavanja korištenjem algoritma temeljnog na detekciji konture

5.6. Ostvareni rezultati

Napravljeno je 5 različitih eksperimenata na 20 uzoraka primjenom oba algoritma što daje broj od ukupnih 200 eksperimenata. Ispitivani objekti razlikuju se po obliku, orijentaciji, veličini i međusobnom položaju. Uvjeti snimanja su poznati te predstavljeni kroz prethodna poglavlja.

Dodatno je izvršeno ispitivanje osjetljivosti na rezoluciju slike u kojem je testirano 40 uzoraka u nesređenim scenama.

5.6.1. Rezultati testiranja algoritma koji se temelji na Houghovim transformacijama i analizi intenziteta obodnih piksela

Uzorci korišteni pri ispitivanju snimljeni su u determiniranim uvjetima osvjetljenja koje je ranije opisano. Ispitivanje se provodilo uz zadani radijus od $r = 75$. Uzeti su uzorci bez preklapanja, uzorci koji su okrenuti, uzorci uz preklapanje te su analizirane scene na kojima je bio postavljen veći broj uzoraka različitog položaja te različitog međusobnog odnosa. Uspješnost prepoznavanja algoritma pri detekciji jednog uzorka u idealnim uvjetima te uvjetima preklapanja prikazan je u tablici 5.5.

Tablica 5.5 Uspješnost prepoznavanja primjenom algoritma temeljenom na generaliziranoj Houghovoj transformaciji i analizi intenziteta obodnih piksela kod izdvojenih uzoraka

	Uzorak	Okrenut uzorak	Uzorak se preklapa
Uspješnost detektiranih uzoraka, %	100	100	50
Uspješnost prepoznatih uzoraka, %	100	100	50
Prosječno vrijeme obrade, s	0,01924	0,01986	0,01732

Iz tablice je vidljivo kako algoritam nema problem s orijentacijom objekta te radi uz 100% učinkovitost. Važno spomenuti da se iskazana vremena procesiranja odnose na slike veličine 240×240 piksela snimljene pri brzini zatvarača 1/15.

Tablica 5.6 Uspješnost prepoznavanja primjenom algoritma temeljenom na generaliziranoj Houghovoj transformaciji i analizi intenziteta obodnih piksela kod nesređenih dvodimenzionalnih scena

	Preklapanje uzoraka	Nesređene scene	
		Izolirani uzorci	Preklopljeni uzorci
Uspješnost detektiranih uzoraka, %	87,5	100	100
Uspješnost prepoznatih uzoraka, %	17,857	100	36,842
Postotak točnosti, %	15,625	100	36,842
Uspješnost prepoznavanja nesređenih scena, %	15,625	68,421	
Ukupna uspješnost prepoznavanja nesređenih scena, %	50,389		
Prosječno vrijeme obrade, s	0,03919	0,19586	

U tablici 5.6 provedeno je ispitivanje na nesređenim scenama ili njihovim dijelovima čiji je cilj praćenje osjetljivosti na preklapanje. U prvom stupcu prikazani su rezultati kod kojih su ispitivani samo dijelovi scena kod kojih imamo preklapanje uzoraka. Veličina ispitivanih slika je 420×420. U drugom stupcu dani su podaci za realne nesređene scene kod kojim nalazimo različit raspored preklopljenih i neprekloplјenih uzoraka. Slike su originalnog formata te imaju veličinu 1920×1200. Utjecaj rezolucije na vrijeme obrade uočljiv je u prosječnom vremenu potrebnom za obradu.

Linearna ovisnost uspješnosti prepoznavanja algoritma u ovisnosti o promjeni rezolucije prikazana je u tablici 5.7. Utjecaj smanjenja rezolucije je ispitivan na realnim nesređenim scenama. Iz tablice je vidljivo kako uspješnost algoritma neznatno manja pri smanjenju rezolucije čak do 50%. Daljnjim smanjenjem uočava se naglo smanjenje postotka učinkovitosti prepoznavanja promatranih objekata.

Tablica 5.7 Uspješnost detekcije i identifikacije u ovisnosti o smanjenju rezolucije

	Smanjenje rezolucije					
	25%		50%		75%	
	Nesređene scene					
	Izolirani uzorci	Preklopljeni uzorci	Izolirani uzorci	Preklopljeni uzorci	Izolirani uzorci	Preklopljeni uzorci
Uspješnost detektiranih uzoraka, %	100	100	100	100	100	94,747
Uspješnost prepoznatih uzoraka, %	100	39,535	100%	28,953	77,421	18,421
Kumulativna uspješnost, %	69,768		64,477		48,176	
Prosječno vrijeme obrade, s	0,08891		0,08140		0,06432	

5.6.2. Rezultati testiranja algoritma koji se temelji na pronalasku kontura

U tablici 5.8 prikazani su rezultati prepoznavanja primjenom algoritma koji se temelji na pronalasku konture. Razvijeni algoritam radi na način gdje je moguće prepoznavanje samo jednog uzorka te je stoga ispitivanje provedeno samo na izoliranim uzorcima koji se razlikuju po orijentaciji ili razini preklapanja. Ispitivani slike uzoraka imaju dimenzije 420×420. Vrijeme potrebno za procesiranje se porastom rezolucije značajno povećava. Ispitani su i uzorci koji se nalaze na scenama slike originalne veličine 1920×1200 međutim zbog prosječnog vremena od 29,86 sekundi ovi rezultati se ne mogu smatrati relevantnima jer algoritam nije razrađen do razine gdje može obrađivati takve nesređene scene. Stoga se iskazano vrijeme ne može uzeti kao značajan parametar.

Tablica 5.8 Uspješnost prepoznavanja primjenom algoritma temeljenom na detekciji konture kod izdvojenih uzoraka

	Uzorak	Okrenut uzorak	Uzorak se preklapa
Uspješnost detektiranih uzoraka, %	100	100	55
Uspješnost prepoznatih uzoraka, %	100	100	55
Prosječno vrijeme obrade, s	4,2978	4,3557	4,0906

Iako se primjenom predloženog algoritma uz pomoć detektirane konture može relativno točno odrediti broj zubi te kutne udaljenosti, on je ograničen na detekciju i identifikaciju samo jednog uzorka. Zanimljiva je vrijednost dana za preklopljene uzorke gdje predloženi algoritam postiže bolje rezultate u usporedbi sa svojim prethodnikom. No važno je napomenuti prethodno izloženu tezu kako je algoritam u ovoj fazi ograničen na prepoznavanje samo jednog objekta te stoga postoji prostor za poboljšanje ovog ograničenja. U testiranim grupama, gdje se uzorci preklapaju, algoritam je izrazito osjetljiv pri čemu pokazuje uspješnost prepoznavanja od samo 10% testiranih uzoraka. Rezultat je poražavajući u usporedbi s prvim predstavljenim algoritmom koji je to činio s postotkom od 50 % za identične ispitne uzorke. Iako pokazuje dobra svojstva prepoznavanja jednog objekta u determiniranim uvjetima i to radi sa 100% učinkovitosti primjena ovog algoritma bila bi odbačena u prvom planu zbog vremena procesiranja. Za uspješnu detekciju i identifikaciju predloženi algoritam koristi 200 puta duže vrijeme od prethodno predstavljene metode koji ima jednaku uspješnost prepoznavanja.

6. ZAKLJUČAK

Rad se bavi razvojem vizijskog sustava za kontrolu kvalitete uložaka grebenaste sklopke. Prvi dio rada opisuje važnost primjene strojnog vida te daje pregled glavnih metoda korištenih pri rješavanju opisanog zadatka. Obrađeno je prvenstveno područje računalnog vida koje se bavi postupcima procesiranja slike te značajkama koje su od važnosti za postupak detekcije i identifikacije.

U drugom dijelu rada predstavljene su dvije algoritamske metode prikladne za robusno prepoznavanje. Obje algoritamske metode su razvijene u besplatnoj i javno dostupnoj biblioteci otvorenog koda OpenCV. Za uspješnost prepoznavanja i lokalizacije, nužan početni uvjet bio je osiguranje uniformnog osvjetljenja. Oba pristupa testirana su na velikom broju slika različite složenosti (u rasponu od jednostavnih slika bez preklapanja uzoraka do slika sa znatnim preklapanjem uzoraka).

Prva algoritamska metoda koristi Houghovu kružnu transformaciju na binarnim slikama kako bi se locirali uzorci, koji se zatim kategoriziraju analizom intenziteta obodnih piksela. Prednosti razvijenog algoritma u danim rezultatima, ogledaju se u vremenu odziva te robusnosti o osvjetljenju, rezoluciji i orijentaciji. Negativna strana je relativna osjetljivost na preklapanje. Na mjestima gdje u većoj mjeri dolazi do preklapanja između ispitnih uzoraka, algoritam detektira visoku vrijednost intenziteta piksela čija širina ne odgovara širini zuba. Prema tome, objekt ne može klasificirati u dostupne kategorije nakon čega ga odbacuje. Istaknuti problem može biti riješen osiguravanjem uvjeta kontrole pri kojem ne bi došlo do preklapanja.

Druga algoritamska metoda temelji se na segmentaciji slike i analizi bloba pri čemu se koriste momenti za lokalizaciju i analiza konture za prepoznavanje. Kao i prethodnik, algoritam pokazuje dobre rezultate ispitivanja kod izoliranih ispitnih uzoraka. Dobra svojstva algoritma se očituju u neosjetljivosti prema osvjetljenju, rezoluciji te orijentaciji. Iako je algoritam zadovoljio u pogledu robusnosti i prepoznavanja danih objekata on ima nekoliko nedostataka. Prvi nedostatak je vrijeme odziva koje je 200 duže u odnosu na prvi algoritam. Za prepoznavanje samo jednog objekta na sceni, algoritmu je u prosjeku potrebno otprilike nešto više od 4 sekunde. Iako pokazuje nešto bolje rezultate od prethodnika, također je osjetljiv na preklapanje. Zadnji nedostatak algoritma je ograničenje u pogledu detekcije više objekata na sceni. Naime, algoritam je u trenutnoj fazi razvoja sposoban izdvojiti samo jednu konturu te nju točno detektirati, pri čemu ostatak scene zanemaruje.

Ispitivanje bi međutim, moglo bi biti napravljeno iterativnim postupkom detekcije. Nakon detekcije prvog objekta najveće konture, detektirani predmet može biti uklonjen te se postupak detekcije može ponoviti za ostale prisutne objekte. Jasno je da ovakav postupak računski intenzivniji.

Dodatno unaprjeđenje može biti ostvareno paralelnim izvršavanjem algoritma primjenom višejezgrenog procesora, pri čemu bi se znatno smanjilo vrijeme procesiranja. Razvoj i testiranje algoritma provedeni su na prijenosnom računalu tvrtke Lenovo (AMD Ryzen 5 2500,2.0Hz, 4MB L2 cache, 8GB RAM) koje posjeduje 64 bitnu arhitekturu te daje zadovoljavajuće vremenske odzive. Međutim, zbog vremenskog ograničenja izrade rada, poboljšanje paralelnim izvršavanjem algoritma nije moglo biti implementirano.

Obzirom na sve izneseno zaključuje se kako oba algoritma zadovoljavaju po pitanju robusnosti prepoznavanja koje se može koristiti pri kontroli uzoraka. Međutim, algoritam temeljen na Houghovoj kružnoj transformaciji i analizi obodnih piksela zadovoljava tražene zahtjeve uz znatno manje vrijeme odziva i računske intenzivnosti. Na kraju se može dati usporedba u pogledu cijene, gdje razvoj ovako izvedenog sustava uz odgovarajuću opremu te predloženi algoritam odgovara svega jednoj desetini cijene dostupnih gotovih rješenja za strojni vid.

LITERATURA

[1] Cognex. Introduction to Machine Vision: A guide to automating process & quality improvements. Sjedinjene Američke države: Cognex Corporation; 2016.

[2] Zuech N. Understanding and Applying Machine Vision. New York: Marcel Dekker;, 2000.

[3] Jordan A. Bildsensoren: So funktionieren sie. Germany; 2017.

[4] Lončarić S. Digitalna obrada slike: Uvod u digitalnu obradu i analizu slike [nastavni materijali iz kolegija]. Zagreb: FER; 2004.

[5] Szelinski R. Computer Vision: Algorithms and applications draft. 2nd ed. Sjedinjene američke države: Springer; 2020.

[6] Džaja B. Digitalna obrada i analiza slike [nastavni materijali iz kolegija]. Split; 2016.

[7] Mostafa G. Digital Image Processing: Digital Image Fundamentals. Egipat: Ain Shams University; 2016.

[8] Miljković O. Image pre-processing tool. Beograd: College of Computer Science, Megatrend University of Belgrade; 2006.

[9] Image Processing 101 Chapter 2.3: Spatial Filters (Convolution). Dynamsoft; 2019. Available from: Image Processing 101 Chapter 2.3: Spatial Filters (Convolution) (dynamsoft.com).

[10] Dewegan S, Sharma A. Image Smoothening and Sharpening using Frequency Domain Filtering Technique. International Journal of Emerging Technologies in Engineering Research. India: IJETER; 2017.

[11] Kaur D, Kaur Y. Various Image Segmentation Techniques: A review. International Journal of Computer Science and Mobile Computing. India: IJCSMC, May, 2014., p. 809-814.

[12] Chapter 4: Segmentation. Available from: https://www.bioss.ac.uk/people/chris/ch4.pdf.

[13] Dass R, Priyanka Devi S. Image Segmentation Techniques- Vol 3. India: Iject; 2012.

[14] Zaitoun N, Aqul MJ. Survey on Image Segmentation Techniques. International Conference in Communication Management and Information Technology. Jordan: Elsevier; 2015.

[15] Nixon MS, Aguado AS. Feature Extraction and Image Processing. 1st edition. Great Britain; Newnes; 2002.

[16] Poobathy D, Manicka Chezian R. Edge Detection Operators: Peak Signal to Noise Ratio Based Comparison. India: IJIGSP; 2014., p. 55-61, DOI: 10.5815/ijigsp.2014.10.07.

[17] Saini S. Arora K. A Study Analysis on the Different Image Segmentation Techniques. International Journal of Information & Computation Technology, ISSN 0974-2239. India: International Research Publications House; 2015.

[18] Staroveški T. Strojni vid u vođenju industrijskog robota [diplomski rad]. Zagreb: FSB; 2007.

[19] Jain R, Kasturi R, Schunck G. Machine Vision, Chapter 5: Edge detection. Sjedinjene američke države: McGraw-Hill Inc; 1995., p. 140-185, ISBN 0-07-032018-7.

[20] Bansal M, Kumar M, Kumar M. 2D Object Recognition Techniques: State-of-the-Art Work. Barcelona: CIMNE; 2020 Feb.

[21] Juneja M, Sandhu PS. Performance Evaluation of Edge Detection Techniques for Images in Spatial Domain. International Journal of Computer Theory and Engineering, ;2009 Dec; 1(5):614-621. doi. 10.7763/IJCTE.2009.V1.100.

[22] Toquica Cáceres HM, Moreno Cálderon JL. Hough and Watershed Transform Algorithms Brief Review. Bogota: Universidad Nacional De Columbia; 2020 March.

[23] Suzuki S, Abe K. Topological structural analysis of digitized binary images by border following. Japan: Academic Press Inc;1985 April., p. 32-46. doi. 10.1016/0734-189X(85)90016-7

[24] OpenCV: About. Available from: https://opencv.org/about/

[25] DMK 23UX174 USB 3.0 monochrome industrial camera. Available from: https://www.theimagingsource.com/products/industrial-cameras/usb-3.0-monochrome/dmk23ux174/

[26] CF-HA Series. Available from: https://www.fujifilm.com/us/en/business/optical-devices/machine-vision-lens/cf-ha-series.

[27] OpenCV: Hough Circle Transform. Available from: https://docs.opencv.org/master/da/d53/tutorial_py_houghcircles.html

[28] OpenCV: Contour Features. Available from: https://docs.opencv.org/master/dd/d49/tutorial_py_contour_features.html

PRILOZI

I. CD-R disc