Факултет инжењерских наука Универзитета у Крагујевцу

Назив студијског програма: Електротехника и рачунарство
Ниво студија: Мастер академске студије
Предмет: Методе формирања и обраде дигиталне слике
Број индекса: 450/2019

Александра Д. Радовић

Издвајање карактеристичних обележја дигиталне слике

Мастер рад

Комисија за преглед и одбрану:
1. др Маријана Гавриловић Божовић, доцент - ментор
2. _____
3. _____

Датум одбране: септембар 2020.
Оцена: _____

У оквиру мастер рада кандидаткиња ће проучавати методе које се користе за издвајање карактеристичних обележја дигиталне слике. Од интереса ће бити обележја ниског нивоа попут ивица, облика, са освртом и на могућности издвајања обележја по боји и текстури. Рад ће се састојати од теоријског дела, где ће кандидаткиња дати преглед резултата из области и објаснити значај правилног издвајања одговарајућих обележја из слике у циљу даље ефикасније класификације, претраживања и сл. Допринос кандидаткиње ће бити и у самостално написаним функцијама које ће вршити издвајање одговарајућих обележја.

Препоручена литература:

[1] Gonzales, C. R., Woods. E. R. *Digital image processing*. (2008). Pearson Education

[2] Gonzales, C. R, Woods E. R., Eddins L. S. *Digital image processing using Matlab*. (2009). Gatesmark Publishing

[3] Поповић, М. *Дигитална обрада слике*. (2006). Академска мисао

Крагујевац, 09.03.2020.

Ментор:

др Маријана Гавриловић Божовић, доцент

_____

Садржај

Abstract

Feature extraction techniques are described in this thesis. The focus is on extraction of low-level features (edges and shapes). This thesis is mainly based on a theoretical review and description of the importance of proper extraction of image features. In addition to the theoretical part, the functions that perform the extraction of appropriate features are also given.

Key words: Digital image, Digital image processing, Edge detection, Gradient operators, Laplace operators, Hough transform, Shape detection

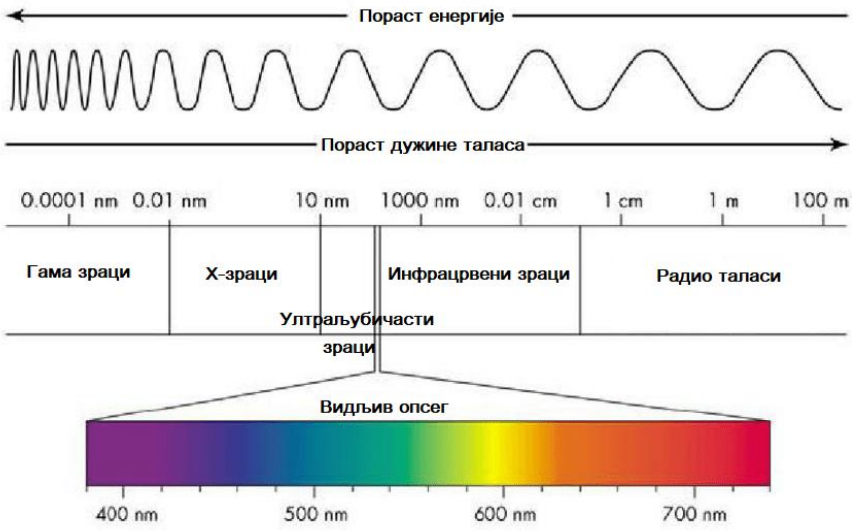
Резиме

У овом раду су описане методе за издвајање карактеристичних обележја слике. Акценат је стављен на издвајање обележја ниског нивоа, односно ивица и облика. Рад се претежно заснива на теоријском прегледу и опису значаја правилног издвајања обележја слике. Поред теоријског дела дате су и функције које врше издвајања одговарајућих обележја.

Кључне речи: Дигитална слика, Обрада дигиталне слике, Детекција ивица у слици, Градијентни оператори, Лапласови оператори, Хафова трансформација, Детекција облика

1. Увод

Чуло вида је најнапредније људско чуло, те стога слике играју најважнију улогу у људској перцепцији. Слика се у људском оку формира на мрежњачи, на задњем зиду ока, тако што светлост претходно улази у око кроз рожњачу и прелама се проласком кроз сочиво. У мрежњачи се налазе специјализоване нервне ћелије – фоторецептори који су осетљиви на светлост и имају способност претварања светлосне енергије у електричне импулсе. За разлику од машина које покривају цео електромагнетни спектар (од гама зрака до радио таласа), људско око је ограничено само на видљив опсег електромагнетног спектра (слика 1) [1].



Слика 1. Електромагнетни спектар

Како би се у даљем раду детаљније објаснио појам и значај дигиталне обраде слике, најпре је потребно објаснити појам саме слике.

Слика, у општем смислу, представља визуелни приказ неког објекта. Она у себи садржи све потребне информације о том објекту које посматрачу омогућавају препознавање самог објекта (нпр. особе која је на слици).

Технички говорећи, слика је континуална функција две променљиве:

$$f(x, y) \tag{1}$$

где су x и y просторне координате. Вредност функције (1) за било који пар координата (x, y) се назива интензитет или ниво сивог на слици у тачки (x, y). Оваква слика се назива још и монохромна слика, односно црно-бела или сива слика. Црно-бела слика представља расподелу интензитета у две димензије.

Слика $f(x, y)$ код које су и просторне координате и интензитет дискретне вредности се назива дигитална монохромна слика. Дигитална слика се састоји од коначног броја елемената од којих сваки има специфичан положај и вредност. Дигитална слика се може представити помоћу матрице чији индекси врста и колона означавају положај посматране тачке на слици, а вредност елемента матрице представља ниво сивог те тачке [2].

1.1. Историја слике

Једна од првих примена дигиталне слике била је у новинској индустрији, када су се слике слале подморском кабловском везом између Лондона и Њујорка. На слици 2 је приказана једна од првих слика пренесена телеграфском везом преко Атлантског океана и репродукована телеграфским штампачем [3].

Слика 2. Слика пренесена подморским каблом 1921. године

Употреба оваквог начина преноса слике скратила је време преноса слике са више од недељу дана на мање од три сата у поређењу са претходним начином преноса слике.

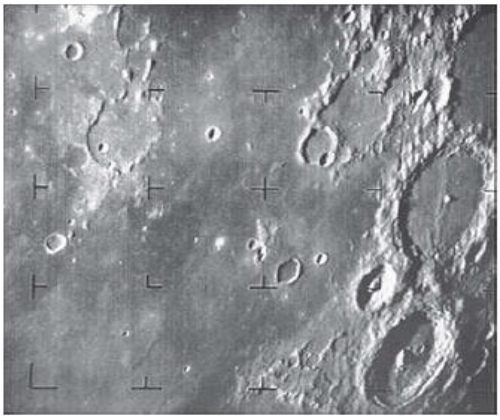
Неки од првих проблема побољшања визуелног квалитета раних дигиталних слика су били повезани са избором процедура штампања и дистрибуцијом нивоа интензитета. Метода која је коришћена за штампање слике 2 је престала да се користи крајем 1921. године и замењена је техником која се базира на репродукцији фотографија направљених од трака, избушених на терминалу за пријем телеграфа. На слици 3 је приказана слика добијена описаном техником [3].

Слика 3. Дигитална слика настала 1922. године са траке пробушене након што су сигнали два пута прешли Антарктик

Рани Бартлејн системи (енг. *Bartlane systems*) су могли да кодирају слике у пет различитих нивоа сиве. 1929. године је број нивоа сиве у којима су се слике кодирале порастао на 15. На слици 4 ће бити приказана дигитална слика из 1929. године пренесена из Лондона до Њујорка кодирана са 15 нијанси сиве [3].

Слика 4. Слика кодирана са 15 нијанси сиве настала 1929. године

Иако се наведени примери односе на дигиталне слике, они нису резултати дигиталне обраде слике, јер не подразумевају обраду слике употребом рачунара. Једна од првих слика која је прошла дигиталну обраду слике у циљу побољшања квалитета је прва слика месеца приказана на слици 5 [3].



Слика 5. Прва слика месеца настала 31. јула 1964. године

Паралелно са применом у сврхе истраживања свемира, технике дигиталне обраде слике су крајем 1960-их и почетком 1970-их почеле да се користе и у медицинске сврхе, посматрања земаљских ресурса и астрономије. Касније се спектар примене обраде дигиталне слике значајно проширио, тако да се данас обрада слике користи и у области индустрије, медицине, биологије, географије, археологије, физике итд.

1.2. Слика у боји

За репрезентацију једног пиксела црно-беле слике је потребан један бајт који може да има вредности од 0 до 255 којима се покривају све нијансе сиве боје. У меморији, црно-бела слика се представља помоћу дводимензионалног низа бајтова, чије су димензије једнаке висини и ширини слике. Овакав низ се назива канал, што значи да црно-бела слика има само један канал који представља интензитет сиве боје.

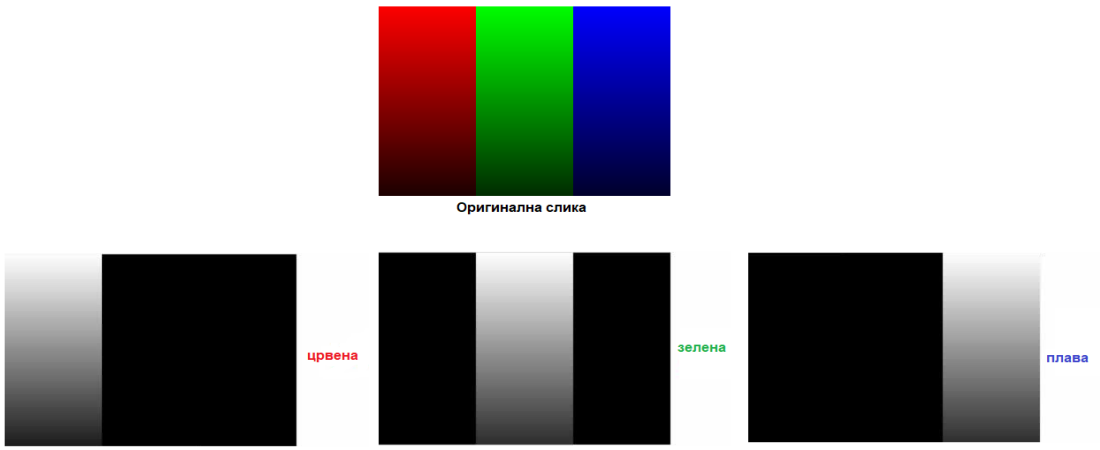
За разлику од црно-беле слике, за репрезентацију пиксела слике у боји су потребна три бајта. Поред интензитета, потребно је памтити и боју, па је за слику у боји потребно више складишног простора од црно-беле слике. За повезивање интензитета са бојом користе се колор системи. Постоји више колор система, а неки од њих су:

- RGB модел (енг. *red, green, blue model*)
- CMYK модел (енг. *cyan, magenta, yellow, black model*)
- HSL/V/B модел (енг. *hue, saturation, lightness/value/brightness model*)

У даљем тексту ће бити описан РГБ колор систем.

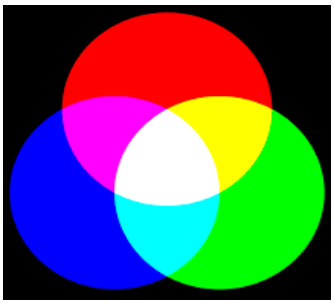
1.2.1. РГБ колор систем

РГБ је колор систем који користи три бајта за сваки пиксел: један бајт за количину црвене боје, други за количину зелене и трећи за количину плаве. На овај начин се добијају три канала: црвени канал, зелени канал и плави канал. РГБ слика се састоји од три независне слике у сивим тоновима које одговарају интензитету црвене, зелене и плаве. Ове три слике се могу одвојено процесирати, а затим рекомбиновати у једну слику коју ће људско око схватити као боју. На слици 6 је приказан начин функционисања РГБ система [4].



Слика 6. Начин функционисања РГБ система

Црвена, зелена и плава су примарне боје у адитивном колор систему, које се могу мешати у различитим пропорцијама за формирање широког спектра боја (слика 7) [5].



Слика 7. Примарне и секундарне боје адитивног колор система

2. Представљање дигиталне слике

Нека је $f(s,t)$ континуална функција слике две континуалне промењљиве, s и t. Оваква функција се конвертује у дигиталну слику узорковањем и квантизацијом. Претпоставимо да се континуална слика узоркује у 2Д низ, (x,y), који садржи M редова и N колона, где су (x,y) дискретне координате:

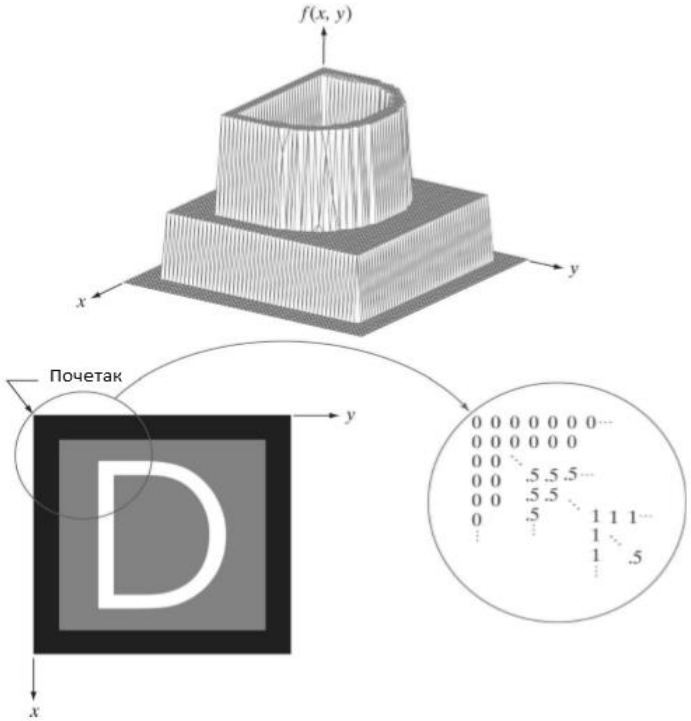
$$x = 0, 1, 2, \cdots, M-1 \tag{2}$$

и

$$y = 0, 1, 2, \cdots, N-1 \tag{3}$$

Тада је, на пример, вредност дигиталне слике у координатном почетку једнака $f(0,0)$, а вредност следеће координате у првом реду је $f(0,1)$ итд. Генерално, вредност дигиталне слике у било којој координати (x,y) се означава као $f(x,y)$, где су x и y целобројне вредности. Област реалне равни означене координатама слике се назива просторни домен, а x и y додељене равни се називају просторне промењљиве или просторне координате.

Као што је у уводном делу већ речено, дигитална слика се може представити помоћу матрице чији индекси врста и колона означавају положај посматране тачке на слици, а вредност елемента матрице представља интензитет или ниво сивог те тачке (слика 8) [2].



Слика 8. Пример представљања дигиталне слике помоћу матрице

На слици 8 је најпре приказан графички приказ функције са две осе које одређују положај у простору и трећом осом која одређује вредности функције f, односно интензитет. Овакав начин представљања слике је користан када се ради са црно-белим сликама чији се елементи представљају као (x,y,z), где су x и y просторне координате, а z вредност функције f у координатама (x,y).

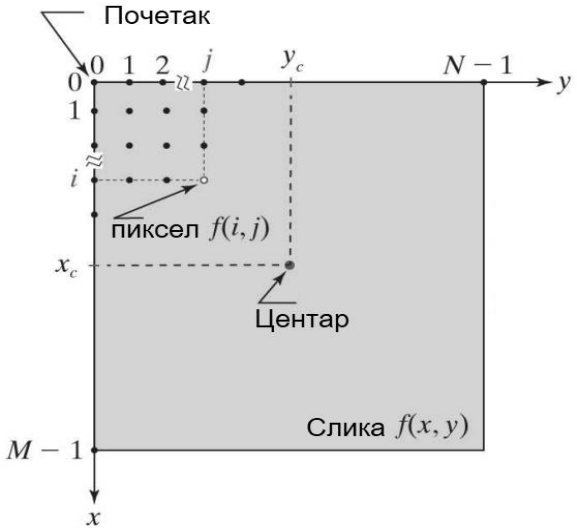
Након тога је дат приказ функције у 2Д простору. Просторне координате остају исте, а трећа координата која је одређивала интензитет функције у тачки се сада нормализује и преводи у вредности између 0 и 1. У примеру са слике 8 вредности интензитета су 0, 0.5 и 1, јер функција слике има само три различите вредности.

Представљање нумеричких вредности функције $f(x,y)$ помоћу низа (матрице) је једноставно. Низ ће бити истих димензија као и сама функција, а вредности низа ће бити једнаке вредностима функције у одговарајућим тачкама.

У облику једначине, матрица $M \times N$ се записује као:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \dots & \dots & \dots & \dots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{bmatrix} \quad (4)$$

Десна страна једначине (4) је дигитална слика представљена помоћу низа реалних бројева. Сваки елемент овог низа се назива елемент слике, односно пиксел. На слици 9 је дат графички приказ низа слике [3].



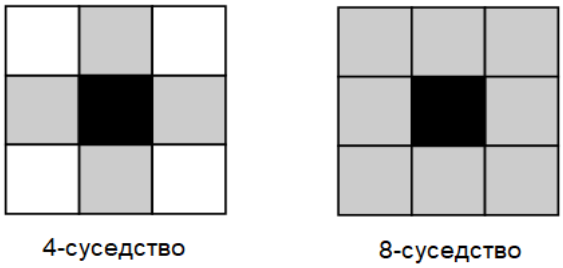
Слика 9. Графички приказ матрице слике

Приказани начин представљања слике се користи за процесирање слике помоћу рачунара.

2.1. Повезаност и суседство пиксела

Повезаност пиксела описује однос између два или више пиксела и испитује се поређењем суседних пиксела у односу на централни пиксел. Суседни пиксели јесу пиксели који додирују посматрани пиксел. Пиксел p чије су координате (x,y) има два хоризонтална и два вертикална суседна пиксела координата $(x+1,y),(x-1,y),(x,y+1),(x,y-1)$. Овакав сет пиксела се назива 4-суседство пиксела p. Поред хоризонталних и вертикалних суседних пиксела, пиксел p има и дијагоналне суседне пикселе координата $(x+1,y+1),(x-1,y-1),(x-1,y+1),(x+1,y-1)$. Ови суседни пиксели, заједно са

претходно описаним 4-суседним пикселима, се називају 8-суседни пиксели [3]. На слици 10 је дат приказ наведених суседства пиксела где је црном бојом означен централни пиксел, а сивом бојом суседни пиксели који се посматрају.



Слика 10. Приказ суседства пиксела

Као што се може видети на слици 10, 4-суседство подразумева посматрање вертикалних и хоризонталних најближих суседних пиксела, док 8-суседство подразумева посматрање и дијагоналних најближих суседних пиксела у односу на посматрани централни пиксел.

У случају када се централни пиксел пореди са четири суседна пиксела, довољно је да има иста својства као један од та четири пиксела да би се сврстао у одговарајућу категорију. Таква повезаност пиксела се назива 4-повезаност пиксела. Уколико се централни пиксел пореди са осам суседних пиксела, таква повезаност се назива 8-повезаност пиксела [2].

Праћење повезаности пиксела се користи за детекцију контура у слици.

3. Резолуција дигиталне слике

Најмањи део дигиталне слике се назива пиксел (енг. *pixel*) који се означава са px. Вредност пиксела је једнака интензитету слике у тачки (x, y). Број пиксела од којих је састављена слика представља просторну резолуцију слике. Резолуција слике се може дефинисати помоћу просторне резолуције и резолуције интензитета, у зависности од тога да ли се посматра број пиксела на слици или број битова по пикселу. У даљем тексту ће бити објашњени појмови просторне резолуције и амплитудске резолуције.

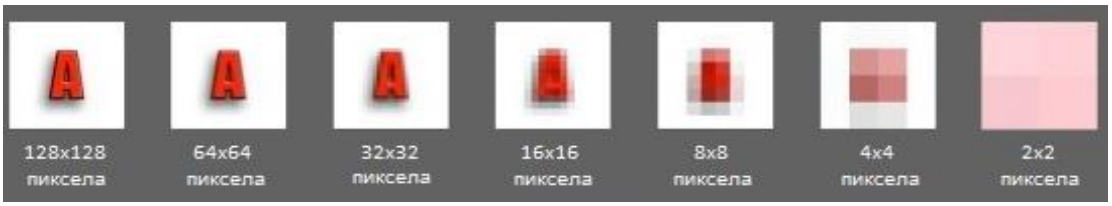
3.1. Просторна резолуција

Просторна резолуција се може дефинисати као мера најмањег уочљивог детаља на слици [4]. Друга дефиниција просторне резолуције јесте укупан број пиксела слике, тј. број пиксела по хоризонтали и вертикали слике.

На пример: Нека је просторна резолуција слике $M \times N$ px. То значи да:

- Слика садржи укупно $M \times N$ пиксела
- Број пиксела по хоризонтали је M
- Број пиксела по вертикали је N

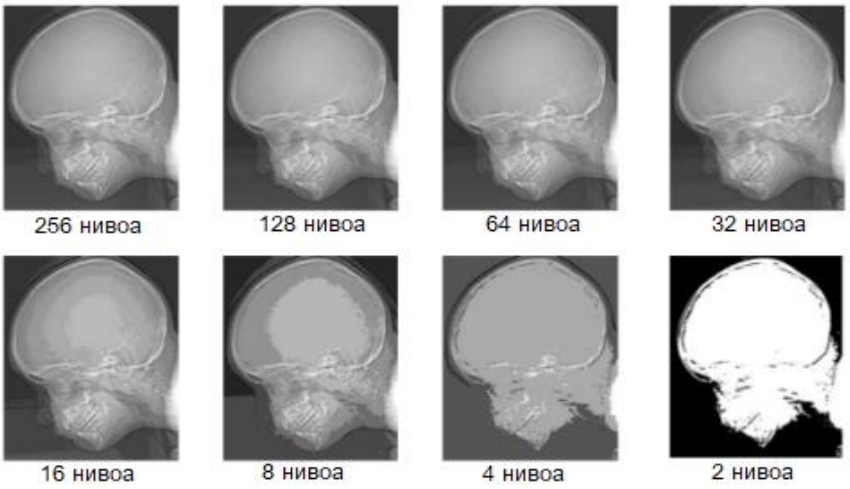
Већа просторна резолуција слике пружа слику са више детаља, односно слику већег квалитета. На слици 11 ће бити дат пример како једна слика може изгледати у различитим просторним резолуцијама [6].



Слика 11. Пример слике у различитим просторним резолуцијама

3.2. Амплитудска резолуција

Амплитудска резолуција (резолуција интензитета) слике се односи на број нивоа интензитета који се користе за представљање слике, односно број битова по пикселу. Број нивоа интензитета је обично степен двојке, најчешће 8 битова. На слици 12 ће бити дат пример како једна слика може изгледати са различитим резолуцијама интензитета [7].



Слика 12. Пример слике са различитим амплитудским резолуцијама

4. Дигитална обрада слике

Историја дигиталне обраде слике је уско повезана са развојем дигиталних рачунара. Дигиталне слике захтевају много складишног простора и рачунарске снаге тако да напредак на пољу обраде дигиталне слике зависи од развоја дигиталних рачунара и технологија које укључују складиштење података, приказ и пренос.

Општа дефиниција обраде слике представља обраду слике као дисциплину у којој је улазни и излазни производ слика [3]. Међутим, често излаз слике могу бити и атрибути издвојени из улазне слике.

Основни циљеви дигиталне обраде слике су:

- Побољшање визуелних карактеристика дигиталне слике
- Процесирање садржаја слике за употребу у аутономним рачунарским системима

У зависности од дефиниције проблема, дигитална обрада слике се може поделити у три нивоа:

1. Обрада ниског нивоа – Подразумева поправку квалитета слике са сензора и припрему за даље процесирање. На пример: изоштравање слике, отклањање шума, побољшавање контраста...
2. Обрада средњег нивоа – Односи се на сегментацију слике и издвајање одговарајућих обележја. На пример: детекција ивица, сегментација, детекција облика...
3. Обрада високог нивоа – Обухвата анализу садржаја слике. На пример: препознавање облика, препознавање садржаја текста...

Процес обраде дигиталне слике зависи од почетног проблема. Кораци који се предузимају у току обраде слике се бирају у односу на дефинисани проблем. На слици 13 ће бити дат преглед основних корака у дигиталној обради слике [3].

Слика 13. Основни кораци у дигиталној обради слике

Кораци који се могу предузети у процесу обраде дигиталне слике приказани на слици 12 су:

- **Прикупљање слика** (енг. *image acquisition*) - поред снимања слике које представља основни и неизоставни корак у оквиру обраде слике, подразумева и предобраду, попут скалирања слике;

- **Побољшање слике** (енг. *image enhancement*) - поступак манипулације сликом, тако да се добије слика бољег квалитета од оригиналне слике; заснива се на субјективном мишљењу гледалаца о томе шта представља „добар“ резултат побољшања;
- **Рестаурација слике** (енг. *image restoration*) - такође спада у поступке унапређења слике, али за разлику од побољшања слике које је субјективно, рестаурација слике је објективна; технике рестаурације се углавном заснивају на математичким или статистичким моделима деградације слике и то су најчешће различите технике филтрирања;
- **Таласи** (енг. *wavelet*) – користе се за компресију дигиталне слике, као и за уклањање шума;
- **Обрада слике у боји** (енг. *color image processing*) - област која добија на значају услед значајног пораста употребе дигиталних слика; боја се користи као основа за издвајање карактеристика слике које су од интереса;
- **Компресија** (енг. *compression*) - подразумева технике за величине смањење меморије потребне за чување слике/видео записа или ширину опсега односно број битова потребних за пренос слике/видео записа;
- **Морфолошко процесирање** (енг. *morphological processing*) - бави се алатима за издвајање компоненти слике које су корисне у представљању и описивању различитих облика;
- **Сегментација** (енг. *segmentation*) - дели слику на њене саставне делове или објекте; ниво поделе и детаља зависи од улазног проблема;
- **Издвајање карактеристика** (енг. *feature extraction*) - скоро увек следи излаз фазе сегментације, што су обично необрађени подаци пиксела који представљају или границу региона или све тачке у самом региону; издвајање карактеристика се састоји од детекције и описа карактеристика; детекција карактеристика се односи на проналажење карактеристика на слици, регија или граница; опис карактеристика издвојеним карактеристикама додељује квантитативне атрибуте;
- **Класификација обрасца слике** (енг. *image pattern classification*) - процес који додељује ознаку објекту на основу описа његових описа карактеристика.

5. Издвајање ивица

Издвајање ивица је један од основних поступака анализе слике који се користи за сегментацију слике и добијање података у областима као што је обрада слике. Основна разлика између алгоритама за обраду слике и алгоритама за анализу слике се огледа у томе што је излаз из система за обраду слике такође слика, док је излаз из система за анализу слике нумерички или симболички опис садржаја слике, а не сама слика.

Поступак анализе слике почиње предобрадом слике неким од поступака за поправку квалитета или рестаурације слике, након чега се из слике издвајају карактеристична обележја потребна за процес издвајања региона, односно сегментацију.

Издвајање ивица је техника анализе слике за проналажење граница објеката слике тако што проналази дисконтинуитет у осветљењу [3]. Основна сврха детекције ивица је поједностављење слике како би се смањила количина података коју треба обрадити и филтрирање свих непотребних информација тако да остану само ивице. На основу карактеристика осветљења или боја, детектор одлучује који је од пиксела слике ивични пиксел. Најпре се врши детектовање ивичних тачака чијим се спајањем добија једна или више ивица, а затим повезивање добијених ивица како би се добила линија која представља контуру објекта.

Један од проблема који се јавља код детекције ивица јесте присуство шумова на слици који доводе до нејасноћа код изгледа слике што отежава детекцију ивица. Због постојања шумова на слици, детектоване ивице се могу поделити у две групе:

- Праве ивице – ивице које стварно припадају некој контури;
- Лажне ивице – ивице које се јављају као последица шума на слици и које не припадају контури.

Током процеса издвајања ивица пролази се кроз неколико корака [8]:

- Филтрирање слике – Обухвата процес отклањања шума и нерелевантних ситних детаља са слике што доводи до побољшања перформанси оператора за издвајање ивица; најчешће се користи Гаусов филтар;
- Детекција ивичних пиксела – У овом кораку се проналазе и наглашавају пиксели код којих постоје значајне промене у вредностима локалног интензитета; овај корак подразумева израчунавање градијента;
- Локализација – Подразумева избор оних пиксела који највероватније припадају ивици из скупа свих детектованих ивичних пиксела.

5.1. Карактеристике дигиталне слике

Као што је већ речено, методе за издвајање ивица у слици се базирају на детекцији оштрих, локалних промена интензитета. Карактеристике слике која доводе до таквих промена интензитета се називају дисконтинуитети слике.

Пиксели у којима долази до нагле промене у интензитету слике се називају ивични пиксели (енг. *edge pixels*), а скупови повезаних ивичних пиксела су ивице или ивични сегменти (енг. *edges*) [3].

Постоји три типа дисконтинуитета слике која су од интереса:

- Изолована тачка – пиксел окружен пикселима много већег или много мањег интензитета у односу на посматрани пиксел;
- Линија – танки ивични сегмент код кога је интензитет позадине или једне од страна линије много већи или много мањи од интензитета линијског сегмента;
- Ивица – скуп повезаних ивичних пиксела који формирају границу између два региона на слици.

5.2. Моделовање ивица

Ивице представљају границе између регија у дигиталној слици и представљају информације од значаја с обзиром на то да одговарају геометријским и физичким променама објекта. На основу разлика у нивоу сивог и интензитету слике, постоји неколико типова ивица:

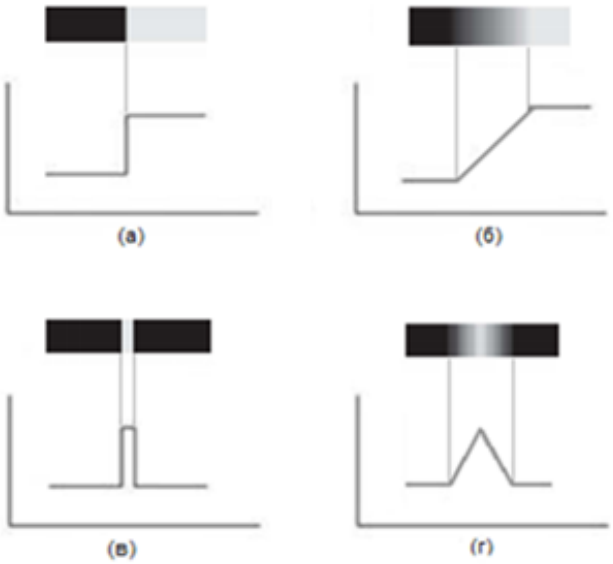
а) Степ ивица (енг. *step edge*) – подразумева идеалну промену једног нивоа интензитета у други на растојању од једног пиксела; пример овакве ивице је компјутерски генерисана слика;

б) Рампа ивица (енг. *ramp edge*) – овај тип ивица имају дигиталне слике са ивицама које су замућене шумом, а чији је степен замућена одређен ограничењима у механизму фокусирања; нагиб рампе је обрнуто сразмеран степену замућења на ивици слике;

в) Линија ивица (енг. *line edge*) – модел ивице који подразумева двоструку идеалну промену са једног нивоа интензитета у други на растојању од једног пиксела; састоји се од две степ ивице (при чему се друга степ ивица може дефинисати као ефекат огледала прве степ ивице), при чему је време прелаза из једне у другу степ ивицу кратко, величине неколико пиксела;

г) Кров ивица (енг. *roof edge*) – модел линија чија је основа, односно ширина кров ивице, одређена дебљином и оштрином линије.

На слици 14 је дат графички приказ наведених типова ивица [9].



Слика 14. Графички приказ типова ивица на слици

На основу начина рачунања разлике интензитета одговарајућих пиксела на слици, тј. структуре матрице кернела, разликују се три типа ивица:

- Хоризонталне ивице
- Вертикалне ивице
- Дијагоналне ивице

На слици 15 је на примеру једне слике приказана разлика у детекцији хоризонталних и вертикалних ивица.

Слика 15. Хоризонталне и вертикалне ивице слике

6. Оператори за издвајање ивица у слици

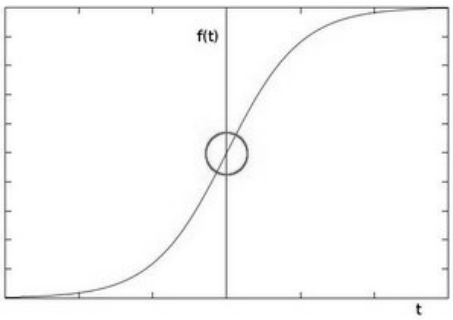
У литератури се може пронаћи много оператора који се користе за издвајање ивица у слици, а већина се може поделити на два типа [10]:

- Градијентни оператори – базирају се на израчунавању првог извода дигиталне слике и проналажењу максимума и минимума првог извода. Оператори за издвајање ивица који спадају у градијентне операторе су:
    - Оператор разлике пиксела
    - Робертсов оператор (енг. *Roberts operator*),
    - Оператор разлике пиксела са размаком,
    - Пруит оператор (енг. *Prewitt operator*),
    - Собелов оператор (енг. *Sobel operator*),
    - Фрај-Ченов оператор (енг, *Frei-Chen operator*),
    - Абдуов оператор (енг. *Abdou operator*),
    - DroG оператор (енг. *Derivative of Gaussian operator*),
    - Канијев оператор (енг. *Canny operator*)
- Лапласови оператори – базирају се на израчунавању другог извода дигиталне слике и тражењу нула другог извода. Примери ових оператора су:
    - Лапласов оператор (енг. *Laplace operator*),
    - Лаплас-Гаусов оператор (енг. *Laplace of Gaussian operator*)

Поред наведених оператора, издвајање ивица у слици је могуће урадити и помоћу модела ивица, односно поређењем са моделом ивице.

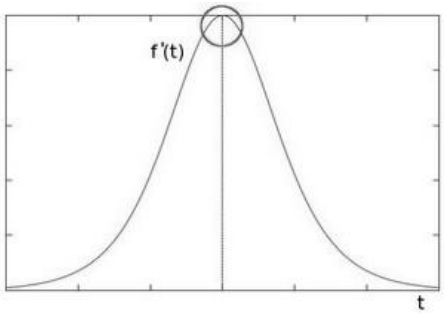
6.1. Употреба извода за детекцију ивица у слици

Графички се ивица једнодимензионалне слике може представити као скок у интензитету (слика 16) [11].



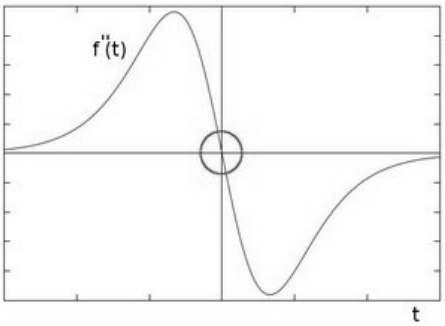
Слика 16. Графички приказ ивице једнодимензионалне слике

Приказани скок ивице се јасније може видети уколико се уради први извод функције где се ивица јавља као максимална вредност функције - максимум функције (слика 17).



Слика 17. Графички приказ првог извода једнодимензионалне слике

На овај начин се ивице у слици могу детектовати локализацијом оних пиксела где је вредност градијента већа у односу на суседне пикселе (или већа од одабране вредности прага). Такође, ивица се може детектовати и помоћу другог извода функције (слика 18).



Слика 18. Графички приказ другог извода једнодимензионалне слике

Као што се може видети са слике 18, ивица се помоћу другог извода може одредити локализацијом оних пиксела где је вредност градијента једнака нули. Међутим, други извод слике неће бити једнак нули само тамо где се појављују ивице слике, већ и на неким мање значајним локацијама. Овај проблем се решава применом филтара.

6.2. Градијентни оператори

Детекција ивица се најчешће врши проналажењем пиксела у чијем окружењу настаје нагла промена нивоа сивог. Тај пиксел је најједноставније пронаћи у случају бинарне слике где се пиксел који припада ивици може дефинисати као црни пиксел са најмање једним суседним белим пикселом или обрнуто [2].

У случају слике са већим бројем нивоа сивог, откривање ивичних пиксела најчешће се врши методама диференцијалне детекције или поређењем са моделом ивице. Диференцијалне методе могу бити засноване на апроксимацији првог или другог извода. У свим случајевима пикселима који припадају ивицама додељује се бела боја, а преосталим црна боја, чиме се добија бинарна слика која се назива мапа ивица.

Приступ заснован на градијенту се назива и маска јер се диференцијалне апроксимације у хоризонталном или вертикалном правцу слике имплементирају помоћу дигиталне маске [12]. Како је ивица дефинисана као део слике у коме се дешавају нагле промене нивоа сивог, први извод слике се користи да се открију максималне и минималне вредности оператора заснованог на градијенту.

Градијентни оператори за издвајање ивица засновани су на дигиталним апроксимацијама првог извода дводимензионалне функције $f(x,y)$ (градијента). Градијент је вектор који показује интензитет и смер максималне брзине промене функције $f(x,y)$:

$$G[f(x,y)] = \nabla f(x,y) = \left[\frac{\partial f}{\partial x}\frac{\partial f}{\partial y}\right]^T \tag{5}$$

У случају дигиталне слике $f[m,n]$, парцијални изводи се апроксимирају количницима коначних прираштаја, односно конволуцијама:

$$G_x[m,n] = f[m,n]^* h_x[m,n] = \sum_{k=-p}^{p}\sum_{l=-p}^{p} h_x[k,l] f[m-k,n-l] \tag{6}$$

$$G_y[m,n] = f[m,n]^* h_y[m,n] = \sum_{k=-p}^{p}\sum_{l=-p}^{p} h_y[k,l] f[m-k,n-l] \tag{7}$$

где су H_x и H_y одговарајуће матрице, или маске, димензије $(2p+1)\times(2p+1)$, које служе за апроксимацију пројекција градијента слике на x и y осу. Амплитуда градијента је онда:

$$G[m,n] = \sqrt{G_x^2[m,n] + G_y^2[m,n]} \tag{8}$$

а аргумент градијента је:

$$\theta_G[m,n] = arctan\left(\frac{G_y[m,n]}{G_x[m,n]}\right) \tag{9}$$

Понекад се за апроксимацију модула градијента користи и формула:

$$G[m,n] = |G_x[m,n]| + \left|G_y[m,n]\right| \tag{10}$$

која је једноставнија за израчунавање.

Одређивањем градијента није завршен процес издвајања ивица. Да би пиксел на локацији $[m,n]$ припадао ивици, потребно је да модул градијента $G[m,n]$ буде већи од неког прага G_T. Тако се добија мапа ивица $E[m,n]$:

$$E(m,n) = \begin{cases} 0, & G[m,n] > G_T \\ 1, & G[m,n] \le G_T \end{cases} \tag{11}$$

Праг G_T се бира тако да се 5% до 10% пиксела са највећим модулима градијента прогласи ивицама. Избор прага се најчешће врши одређивањем кумулативног хистограма градијентне слике $[m, n]$ [2].

Сви градијентни оператори се састоје од конволуционих кернела (H_x и H_y), односно филтара, који се могу применити одвојено на улазну слику како би се добила одвојена мерења градијента за сваку оријентацију (G_x и G_y). Након тога је потребно помоћу добијених вредности пронаћи амплитуду градијента у свакој тачки и оријентацију тог градијента. Амплитуда градијента се рачуна употребом формуле (8), али се често користи и апроксимација (10) која је бржа за рачунање [13]. Угао оријентације ивице се одређује помоћу формуле (9).

У наредном делу овог рада ће бити дат преглед оператора који се користе за издвајање ивица у слици.

6.2.1. Оператор разлике пиксела

Оператор разлике пиксела представља најједноставнију апроксимацију компонената градијента формирањем разлика осветљености пиксела дуж редова и колона слике. Разлике осветљености су позитивне ако осветљеност пиксела у слици расте са лева на десно и одозго на доле [2]:

$$G_x[m, n] = f[m, n] - f[m, n + 1] \tag{12}$$

$$G_y[m, n] = f[m, n] - f[m + 1, n] \tag{13}$$

Оператор разлике пиксела се састоји од два конволуциона кернела:

$$H_x = [1 \quad -1] \tag{14}$$

$$H_y = \begin{bmatrix} -1 \\ +1 \end{bmatrix} \tag{15}$$

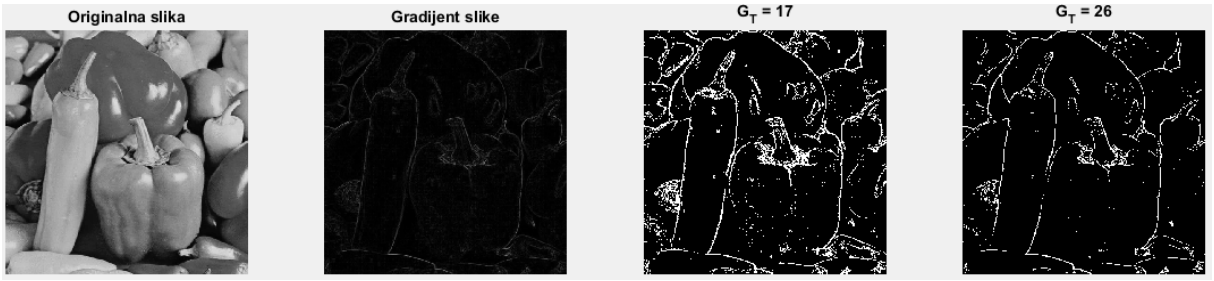
Оператор разлике пиксела се добро показује приликом одређивања положаја оштрих ивица, али лоше приликом одређивања ивица слабијег интензитета – даје лошу локализацију. Још једна лоша карактеристика овог оператора јесте велика осетљивост на шум што доводи до лоше дефинисаности граница региона на слици.

На слици 19 је дат приказ начина имплементације оператора разлике пиксела у Матлабу.

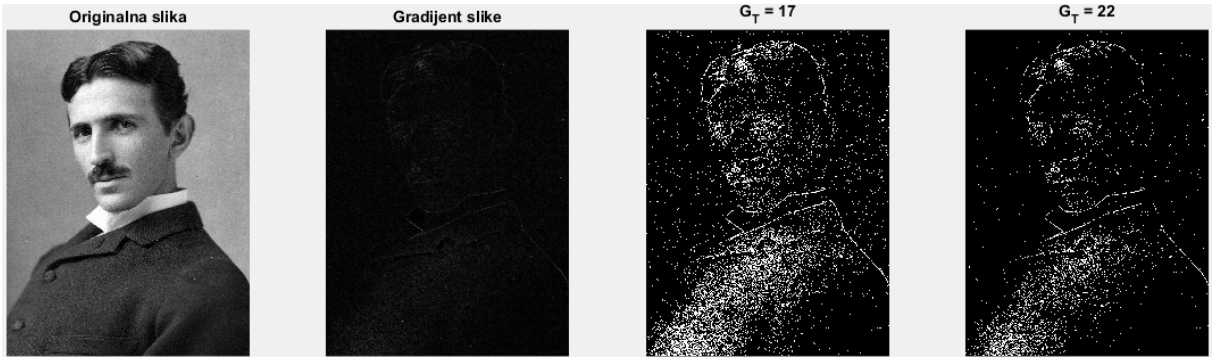
```
Hx = [1 -1];
Hy = [-1; 1];

for i=1:size(inputImage, 1)-1
    for j=1:size(inputImage, 2)-1
        Gx = sum(sum(Hx.*inputImage(i, j:j+1)));
        Gy = sum(sum(Hy.*inputImage(i:i+1, j)));

        gradient(i,j) = sqrt(Gx.^2 + Gy.^2);
    end
end
```

Слика 19. Имплементација оператора разлике пиксела у Матлабу

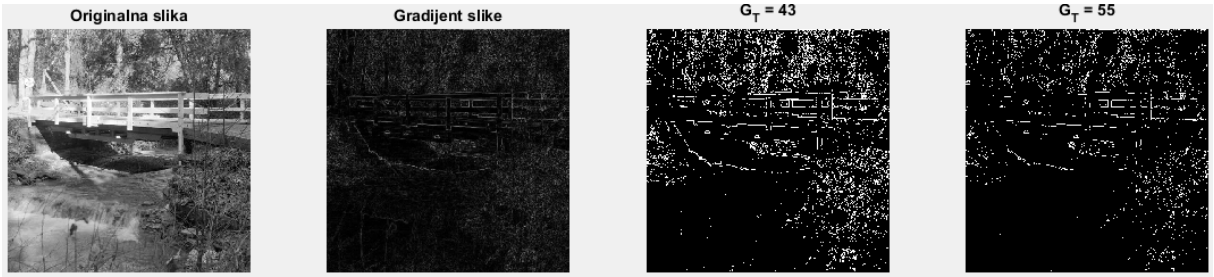
На сликама испод (20-24) је приказан рад овог оператора на сликама различитих карактеристика и постављеном вредношћу прага G_T тако да се 10% и 5% пиксела са највећим градијентом прогласи ивицама.



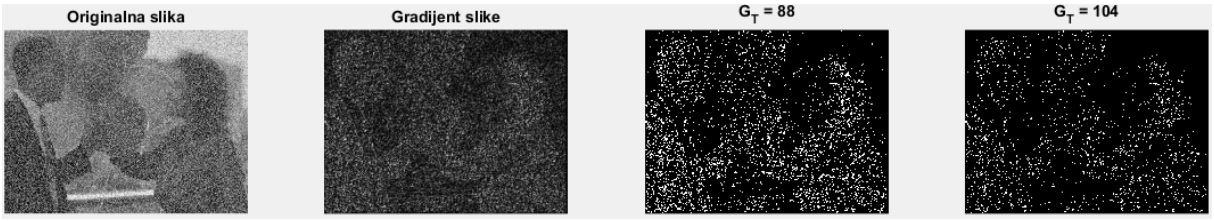
Слика 20. Приказ рада оператора разлике пиксела на слици ивица оштрог интензитета



Слика 21. Приказ рада оператора разлике пиксела на слици са шумом



Слика 22. Приказ рада оператора разлике пиксела на слици са доста детаља



Слика 23. Приказ рада оператора разлике пиксела на слици са доста шума

Слика 24. Приказ рада оператора разлике пиксела на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, оператор разлике пиксела најбоље резултате даје на слици оштрих ивица са мало шума. Лошије резултате даје на слици са доста детаља и слици са ивицама слабијег интензитета. На сликама са шумом је издвојено много лажних ивица што на примеру показује осетљивост овог оператора на шум поменуту приликом описивања оператора.

6.2.2. Робертсов оператор

Робертсов оператор је једноставни оператор који спада у групу градијентних оператора за издвајање ивица, а може се користити и за изоштравање слике. Овај оператор издваја дијагоналне ивице, тј. ивице под угловима од 45° и 135° јер се разлике осветљености пиксела формирају помоћу једначина које рачунају разлику интензитета између централног пиксела и дијагоналних суседа:

$$G_x[m,n] = f[m,n] - f[m+1,n+1] \tag{16}$$

$$G_y[m,n] = f[m,n+1] - f[m+1,n] \tag{17}$$

Оператор се састоји од пара 2×2 конволуционих кернела:

$$H_x = \begin{bmatrix} 0 & -1 \\ +1 & 0 \end{bmatrix} \tag{18}$$

$$H_y = \begin{bmatrix} -1 & 0 \\ 0 & +1 \end{bmatrix} \tag{19}$$

Угао оријентације ивице се одређује помоћу формуле:

$$\theta = arctan\left(\frac{G_y}{G_x}\right) - \frac{3\pi}{4} \tag{20}$$

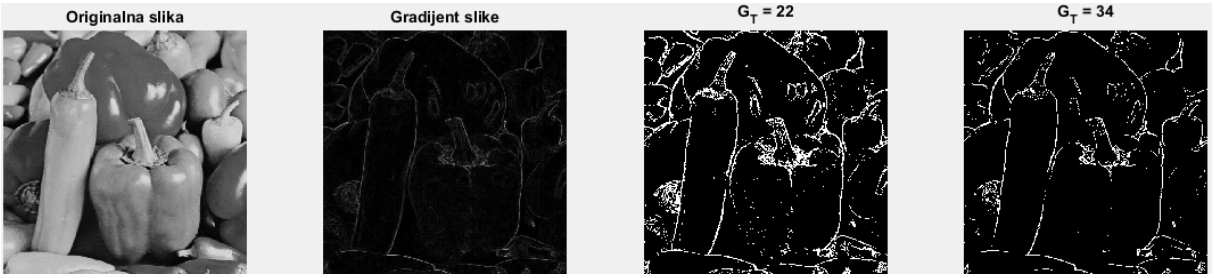
Робертсов оператор проналази ивичне пикселе и не даје информације о оријентацији ивице. Најбоље ради са бинарним сликама. Главна предност овог оператора јесте брзина прорачуна, док су највећи недостаци велика осетљивост на шум због малог кернела и слаба реакција на праве ивице осим ако нису веома оштре. Из тог разлога се у мапи ивица појављују лажне ивице. Како би се овај проблем лажних ивица ублажио, потребно је одабрати већу вредност за праг G_T што доводи до губљења правих ивица ниског интензитета [2].

На слици 25 је дат приказ начина имплементације Робертсовог оператора у Матлабу.

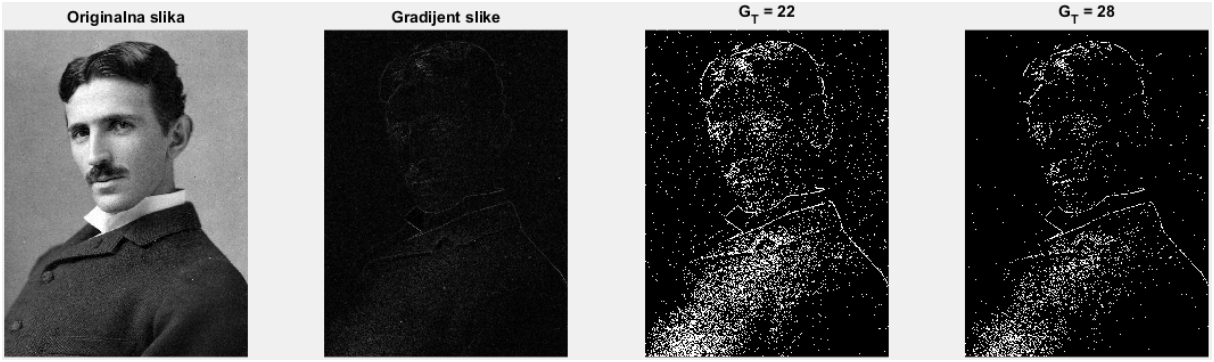
```
Hx = [0 -1; 1 0];
Hy = [-1 0; 0 1];

for i=1:size(inputImage, 1)-1
    for j=1:size(inputImage, 2)-1
        Gx = sum(sum(Hx.*inputImage(i:i+1, j:j+1)));
        Gy = sum(sum(Hy.*inputImage(i:i+1, j:j+1)));

        gradient(i,j) = sqrt(Gx.^2 + Gy.^2);
    end
end
```

Слика 25. Имплементација Робертсовог оператора у Матлабу

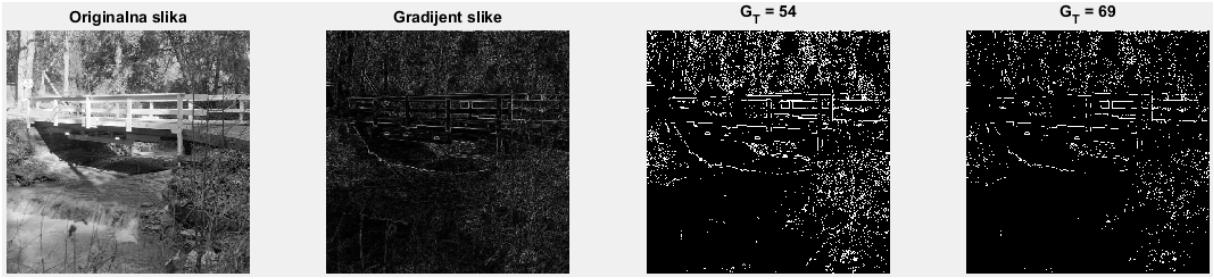
На сликама испод (26-30) је приказан рад овог оператора на сликама различитих карактеристика и постављеном вредношћу прага G_T тако да се 10% и 5% пиксела са највећим градијентом прогласи ивицама.



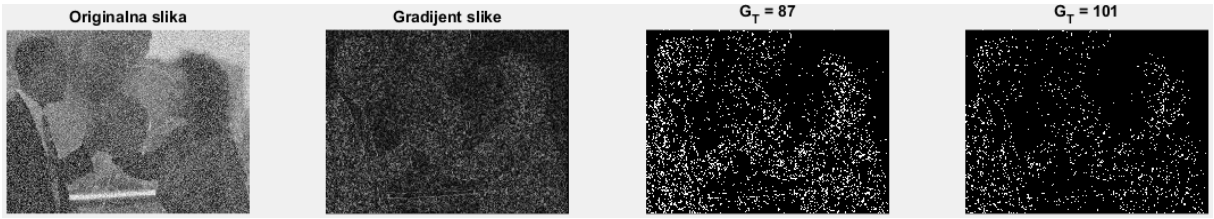
Слика 26. Приказ рада Робертсовог оператора на слици ивица оштрог интензитета



Слика 27. Приказ рада Робертсовог оператора на слици са шумом



Слика 28. Приказ рада Робертсовог оператора на слици са доста детаља



Слика 29. Приказ рада Робертсовог оператора на слици са доста шума

Слика 30. Приказ рада Робертсовог оператора на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, Робертсов оператор се најбоље показао на слици оштрих ивица, док је лошије резултате дао на сликама са ивицама слабијег интензитета и сликама са шумом. На сликама са шумом је и овај оператор детектовао доста лажних ивица као и претходно описани оператор разлике пиксела.

6.2.3. Оператор разлике пиксела са размаком

Оператор разлике пиксела са размаком даје мало боље резултате приликом издвајања ивица у слици у поређењу са претходно описаним Робертсовим оператором. Код овог оператора се разлике осветљености пиксела дуж редова и колона формирају помоћу једначина:

$$G_x[m,n] = f[m,n+1] - f[m,n-1] \tag{21}$$

$$G_y[m,n] = f[m-1,n] - f[m+1,n] \tag{22}$$

Оператор се састоји од пара 3×3 конволуционих кернела:

$$H_x = \begin{bmatrix} 0 & 0 & 0 \\ +1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} \tag{23}$$

$$H_y = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & +1 & 0 \end{bmatrix} \tag{24}$$

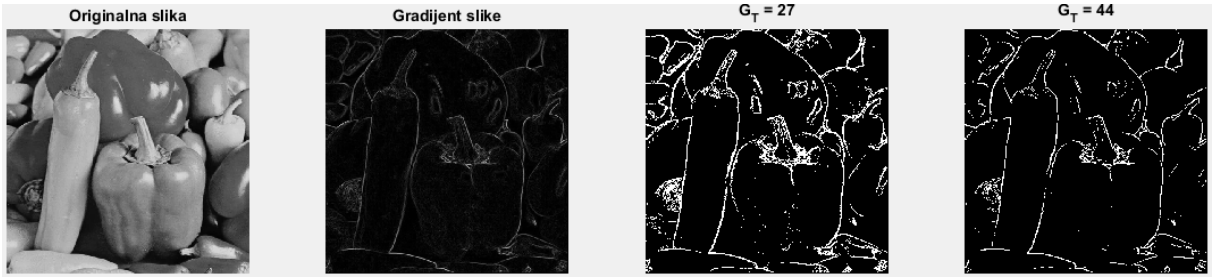
Као и код Робертсовог оператора, предност овог оператора јесте брзина прорачуна, док су највећи недостаци велика осетљивост на шум због малог кернела и слаба реакција на праве ивице осим ако нису веома оштре [2]. Проблем појаве лажних ивица се такође решава одабиром веће вредности прага G_T.

На слици 31 је дат приказ начина имплементације оператора разлике пиксела са размаком у Матлабу.

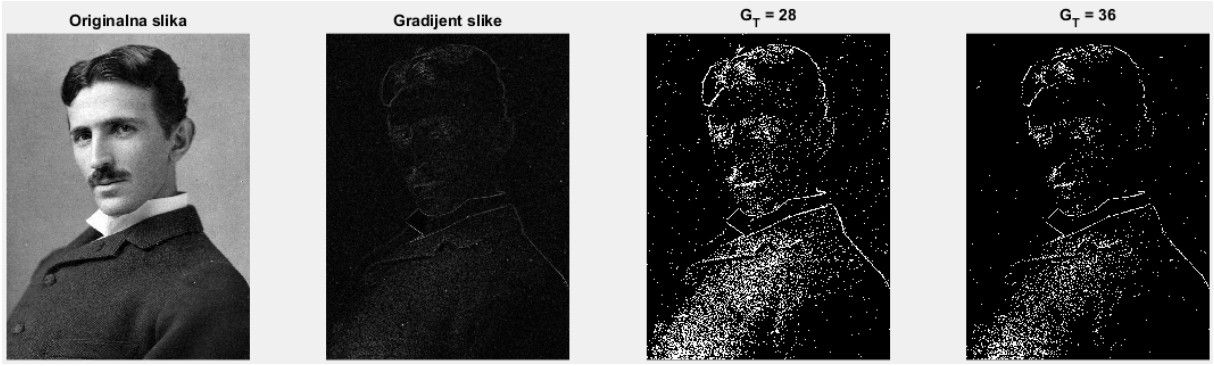
```
Hx = [0 0 0; 1 0 -1; 0 0 0];
Hy = [0 -1 0; 0 0 0; 0 1 0];

for i=1:size(inputImage, 1)-2
    for j=1:size(inputImage, 2)-2
        Gx = sum(sum(Hx.*inputImage(i:i+2, j:j+2)));
        Gy = sum(sum(Hy.*inputImage(i:i+2, j:j+2)));

        gradient(i,j) = sqrt(Gx.^2 + Gy.^2);
    end
end
```

Слика 31. Имплементација оператора разлике пиксела са размаком у Матлабу

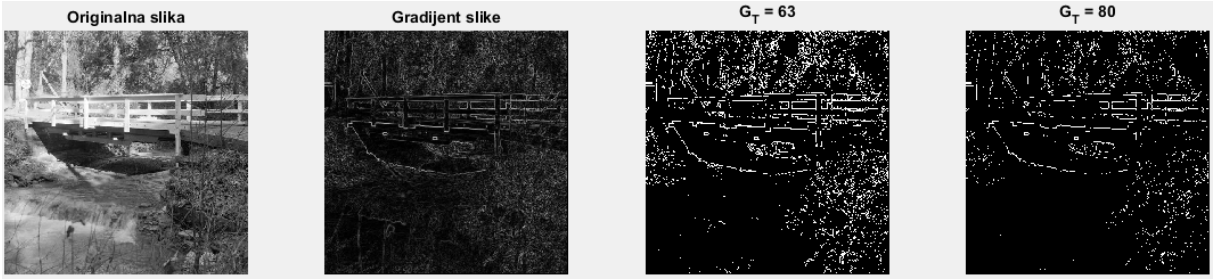
На сликама испод (32-36) је приказан рад овог оператора на сликама различитих карактеристика и постављеном вредношћу прага G_T тако да се 10% и 5% пиксела са највећим градијентом прогласи ивицама.



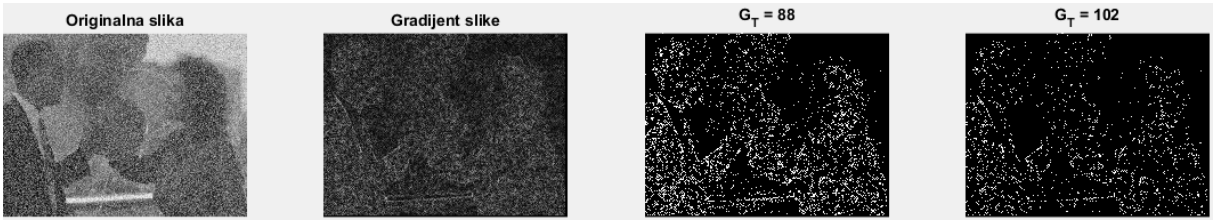
Слика 32. Приказ рада оператора разлике пиксела са размаком на слици ивица оштрог интензитета



Слика 33. Приказ рада оператора разлике пиксела са размаком на слици са шумом



Слика 34. Приказ рада оператора разлике пиксела са размаком на слици са доста детаља



Слика 35. Приказ рада оператора разлике пиксела са размаком на слици са доста шума

Слика 36. Приказ рада оператора разлике пиксела са размаком на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, оператор разлике пиксела са размаком

најбоље резултате даје на слици оштрих ивица. Лошије резултате бележи на слици са ивицама слабијег интензитета и слици са доста детаља. И овај оператор, као и претходно описани оператори, детектује много лажних ивица на сликама са шумом.

6.2.4. Пруит оператор

Проблем са негативним утицајем шума код претходно описаних оператора се делимично може елиминисати применом просторног усредњавања по ортогоналној координати приликом формирања разлике осветљености по једној координати на начин приказан испод:

$$H_x = \frac{1}{K+2}\begin{bmatrix} +1 & 0 & -1 \\ +K & 0 & -K \\ +1 & 0 & -1 \end{bmatrix} \tag{25}$$

$$H_y = \frac{1}{K+2}\begin{bmatrix} -1 & -K & -1 \\ 0 & 0 & 0 \\ +1 & +K & +1 \end{bmatrix} \tag{26}$$

У наведеним једначинама (25) и (26) K је слободан параметар [2]. Просторно усредњавање се извршава тако што се замени вредност сваког пиксела тежинском средњом вредношћу локалног суседства тог пиксела. Просторне маске су најчешће квадратног облика са непарним бројем елемената. Техника просторног усредњавања се најчешће користи ради уклањања шума из слике.

Један од оператора код кога је примењено ово побољшање је Пруит оператор. Овај оператор спада у групу градијентних оператора и користи се за детектовање вертикалних и хоризонталних ивица у слици. Оператор се састоји од пара 3×3 конволуционих кернела:

$$H_x = \frac{1}{3}\begin{bmatrix} +1 & 0 & -1 \\ +1 & 0 & -1 \\ +1 & 0 & -1 \end{bmatrix} \tag{27}$$

$$H_y = \frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ +1 & +1 & +1 \end{bmatrix} \tag{28}$$

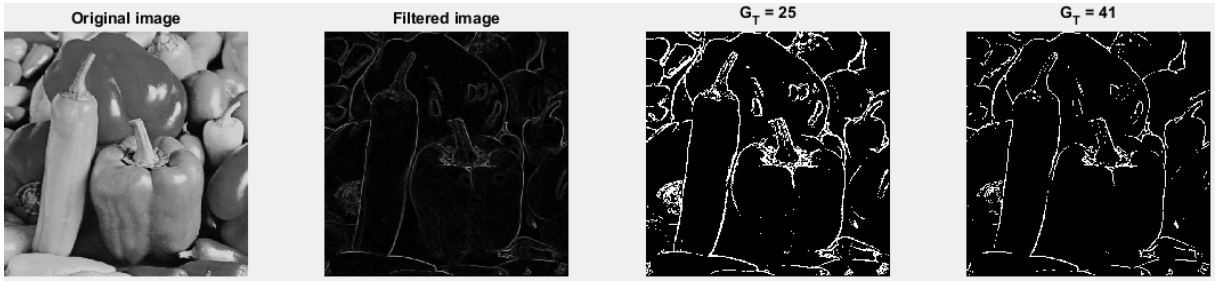
Код Пруит оператора градијент дијагоналних ивица је једнак градијенту ивице у смеру координатне осе помноженим фактором 0.94.

На слици 37 је дат приказ начина имплементације Пруит оператора у Матлабу.

```
Hx = [1 0 -1; 1 0 -1; 1 0 -1];
Hy = [-1 -1 -1; 0 0 0; 1 1 1];
for i=1:size(inputImage, 1)-2
    for j=1:size(inputImage, 2)-2
        Gx = sum(sum(Hx.*inputImage(i:i+2, j:j+2)))/3;
        Gy = sum(sum(Hy.*inputImage(i:i+2, j:j+2)))/3;
        gradient(i,j) = sqrt(Gx.^2 + Gy.^2);
    end
end
```

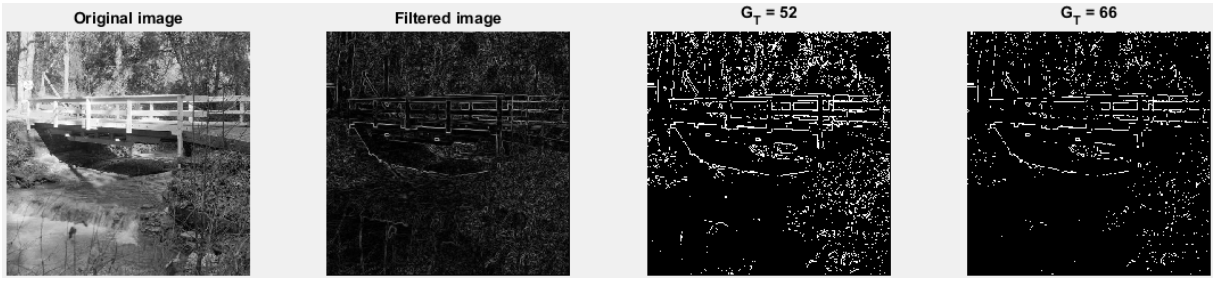
Слика 37. Имплементација Пруит оператора у Матлабу

На сликама испод (38-42) је приказан рад овог оператора на сликама различитих карактеристика и постављеном вредношћу прага G_T тако да се 10% и 5% пиксела са највећим градијентом прогласи ивицама.

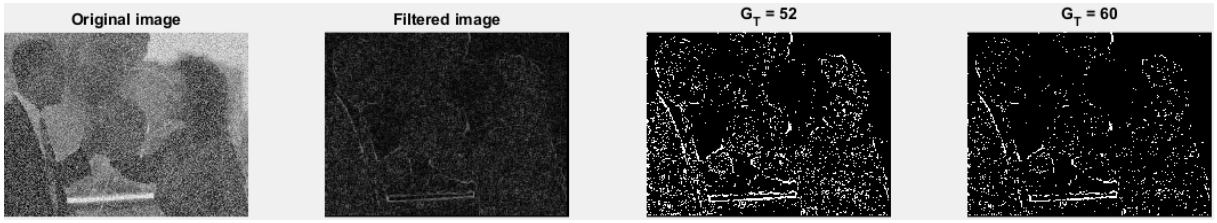


Слика 38. Приказ рада Пруит оператора на слици ивица оштрог интензитета

Слика 39. Приказ рада Пруит оператора на слици са шумом



Слика 40. Приказ рада Пруит оператора на слици са доста детаља



Слика 41. Приказ рада Пруит оператора на слици са доста шума

Слика 42. Приказ рада Пруит оператора на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, Пруит оператор се добро показао на слици оштрих ивица. Нешто лошије резултате бележи на слици са ивицама слабијег интензитета и слици са доста детаља. Слично као и претходно описани оператори, и Пруит оператор детектује много лажних ивица на сликама са шумом.

6.2.5. Собелов оператор

Собелов оператор је један од најчешће коришћених оператора за детекцију ивица и даје мало боље резултате у односу на Пруит оператор. Састоји се од пара 3×3 конволуционих кернела:

$$H_x = \frac{1}{4}\begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \tag{29}$$

$$H_y = \frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \tag{30}$$

Као што се може видети и из конволуционих кернела, вредност параметра K код Собеловог оператора је 2. На тај начин се већи значај даје пикселима лево и десно, односно изнад и испод централног пиксела у односу на остале пикселе који се налазе у суседству посматраног пиксела.

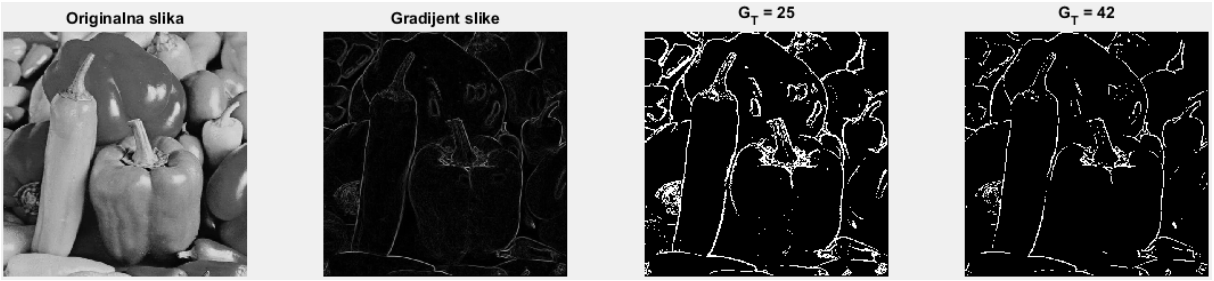
Код Собеловог оператора градијент дијагоналних ивица је једнак градијенту ивице у смеру координатне осе помноженим фактором 1.06 [2].

На слици 43 је дат приказ начина имплементације Собеловог оператора у Матлабу.

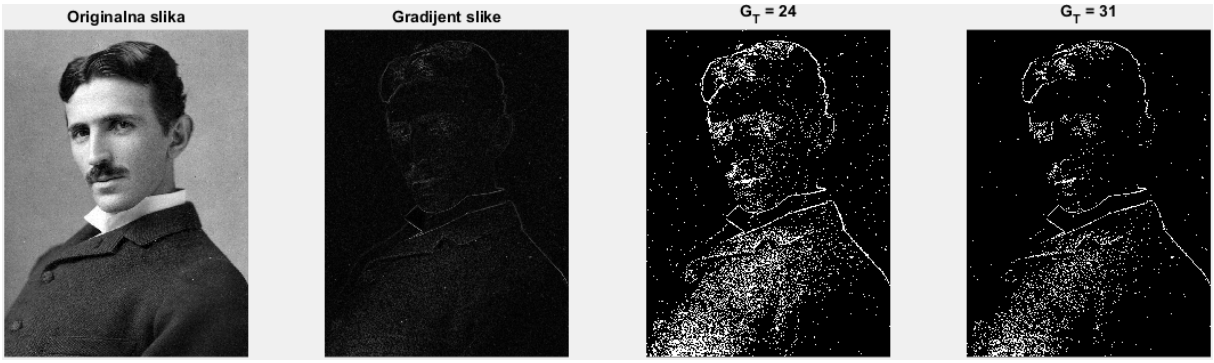
```
Hx = [1 0 -1; 2 0 -2; 1 0 -1];
Hy = [-1 -2 -1; 0 0 0; 1 2 1];

for i=1:size(inputImage, 1)-2
    for j=1:size(inputImage, 2)-2
        Gx = sum(sum(Hx.*inputImage(i:i+2, j:j+2)))/4;
        Gy = sum(sum(Hy.*inputImage(i:i+2, j:j+2)))/4;

        gradient(i,j) = sqrt(Gx.^2 + Gy.^2);
    end
end
```

Слика 43. Имплементација Собеловог оператора у Матлабу

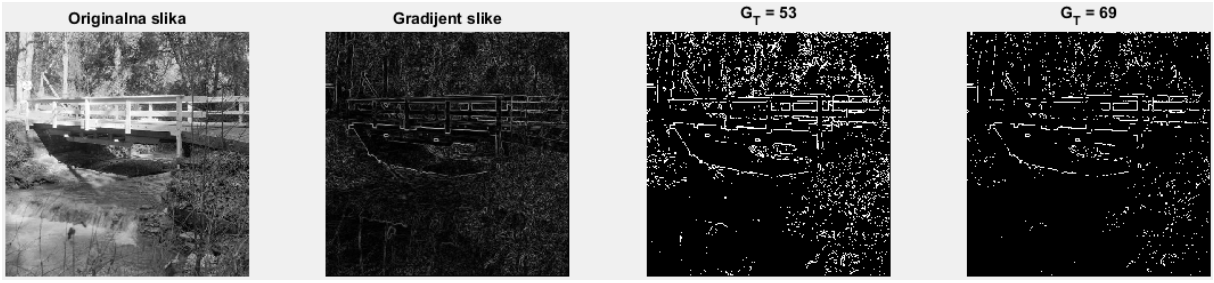
На сликама испод (44-48) је приказан рад овог оператора на сликама различитих карактеристика и постављеном вредношћу прага G_T тако да се 10% и 5% пиксела са највећим градијентом прогласи ивицама.



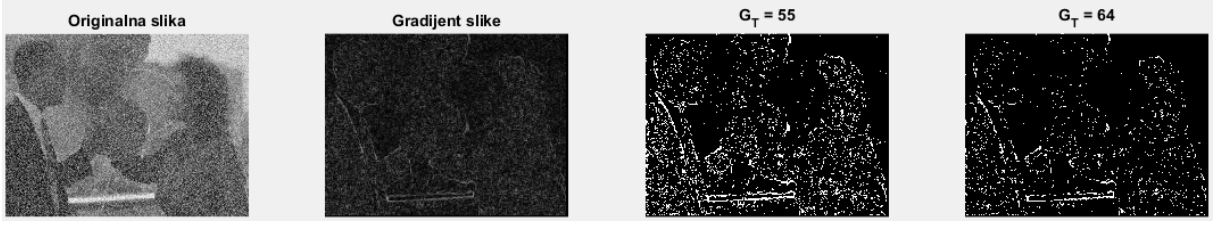
Слика 44. Приказ рада Собеловог оператора на слици ивица оштрог интензитета



Слика 45. Приказ рада Собеловог оператора на слици са шумом



Слика 46. Приказ рада Собеловог оператора на слици са доста детаља



Слика 47. Приказ рада Собеловог оператора на слици са доста шума

Слика 48. Приказ рада Собеловог оператора на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, Собелов оператор се добро показао на слици оштрих ивица. Нешто лошије резултате даје на слици са доста детаља и слици са ивицама слабијег интензитета. Слично као и Пруит оператор, детектује доста лажних ивица на сликама са шумом.

6.2.6. Фрај-Ченов оператор

Још један оператор из групе побољшаних градијентних оператора јесте Фрај-Ченов оператор код кога је вредност параметра K постављена на $\sqrt{2}$. Конволуциони кернели овог оператора су:

$$H_x = \frac{1}{2+\sqrt{2}}\begin{bmatrix} +1 & 0 & -1 \\ +\sqrt{2} & 0 & -\sqrt{2} \\ +1 & 0 & -1 \end{bmatrix} \tag{31}$$

$$H_y = \frac{1}{2+\sqrt{2}}\begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ +1 & +\sqrt{2} & +1 \end{bmatrix} \tag{32}$$

За разлику од Пруит и Собеловог оператора, овај оператор даје једнаку вредност градијента за хоризонталне, вертикалне и дијагоналне ивице.

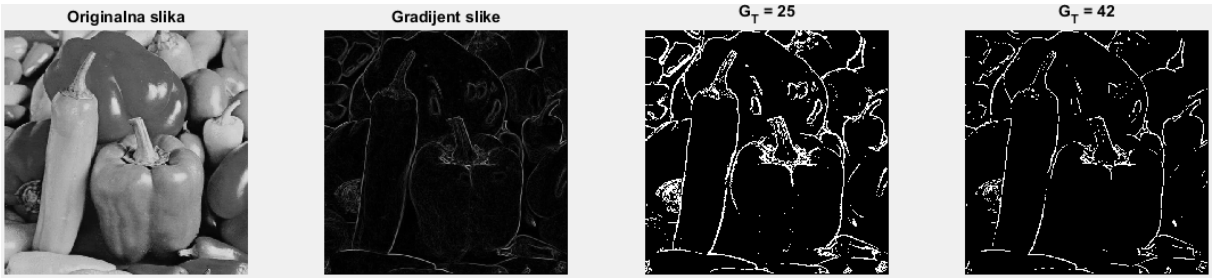
Сва три претходно описана оператора (Пруит, Собел и Фрај-Ченов) дају вредност нула ако се примене на регион униформне осветљености [2]. Такође, сва три оператора се показују знатно боље у погледу лажних ивица и утицаја шума на издвајање ивица у слици у поређењу са операторима који су засновани на разликама.

На слици 49 је дат приказ начина имплементације Фрај-Ченовог оператора у Матлабу.

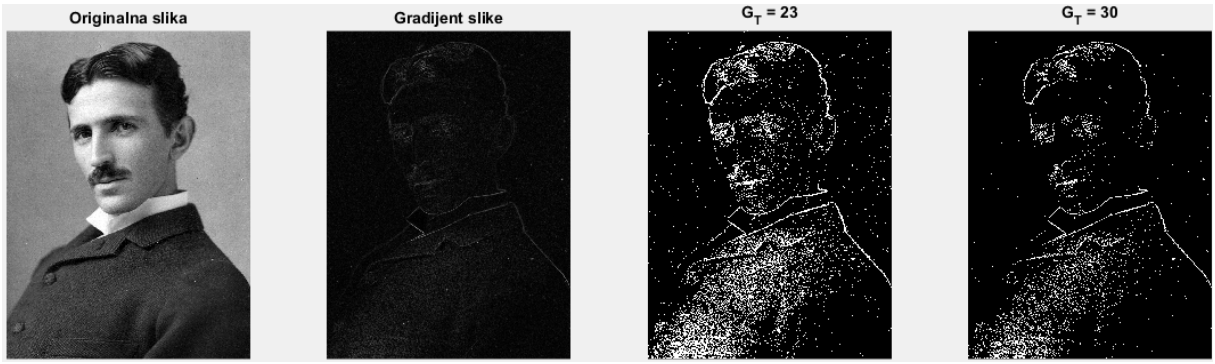
```
Hx = [1 0 -1; sqrt(2) 0 -sqrt(2); 1 0 -1];
Hy = [-1 -sqrt(2) -1; 0 0 0; 1 sqrt(2) 1];

for i=1:size(inputImage, 1)-2
    for j=1:size(inputImage, 2)-2
        Gx = sum(sum(Hx.*inputImage(i:i+2, j:j+2)))/(2+sqrt(2));
        Gy = sum(sum(Hy.*inputImage(i:i+2, j:j+2)))/(2+sqrt(2));

        gradient(i,j) = sqrt(Gx.^2 + Gy.^2);
    end
end
```

Слика 49. Имплементација Фрај-Ченовог оператора у Матлабу

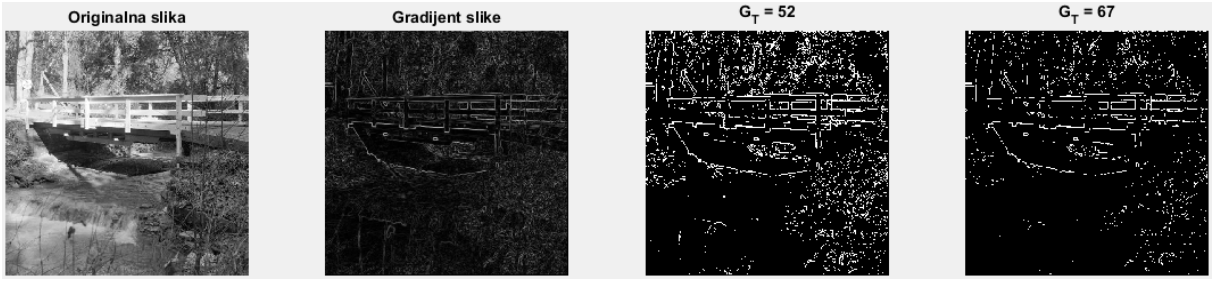
На сликама испод (50-54) је приказан рад овог оператора на сликама различитих карактеристика и постављеном вредношћу прага G_T тако да се 10% и 5% пиксела са највећим градијентом прогласи ивицама.



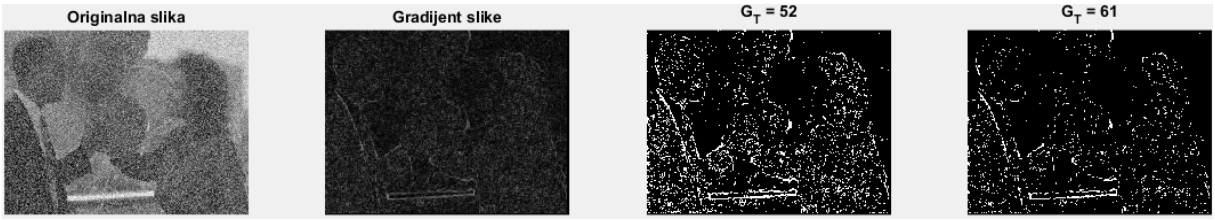
Слика 50. Приказ рада Фрај-Ченовог оператора на слици ивица оштрог интензитета



Слика 51. Приказ рада Фрај-Ченовог оператора на слици са шумом



Слика 52. Приказ рада Фрај-Ченовог оператора на слици са доста детаља



Слика 53. Приказ рада Фрај-Ченовог оператора на слици са доста шума

Слика 54. Приказ рада Фрај-Ченовог оператора на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, Фрај-Ченов оператор даје сличне резултате Собеловом оператору на свим сликама. Добро се показује на слици са оштрим ивицама, лошије на слици са доста детаља и слици са ивицама слабијег интензитета. И овај оператор детектује доста лажних ивица на сликама са шумом.

6.2.7. Абдуов оператор

Поред просторног усредњавања које је примењено за елиминисање утицаја шума приликом издвајања ивица, додатно потискивање шума се може остварити

проширивањем локалног суседства на коме се израчунава апроксимација градијента, односно проширивањем матрице кернела [2]. Оператори код којих је имплементирано потискивање шума проширивањем локалног суседства се називају бокскар оператори.

Један од оператора који припадају бокскар операторима јесте Абдуов оператор који се назива и пирамидални оператор због структуре кернела. Код овог оператора тежински коефицијенти опадају како расте удаљеност од централног пиксела. Абдуов оператор се састоји од пара 7×7 конволуционих кернела:

$$H_x = \frac{1}{34}\begin{bmatrix} +1 & +1 & +1 & 0 & -1 & -1 & -1 \\ +1 & +2 & +2 & 0 & -2 & -2 & -1 \\ +1 & +2 & +3 & 0 & -3 & -2 & -1 \\ +1 & +2 & +3 & 0 & -3 & -2 & -1 \\ +1 & +2 & +3 & 0 & -3 & -2 & -1 \\ +1 & +2 & +2 & 0 & -3 & -2 & -1 \\ +1 & +1 & +1 & 0 & -1 & -1 & -1 \end{bmatrix} \quad (33)$$

$$H_y = \frac{1}{34}\begin{bmatrix} -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -2 & -2 & -2 & -2 & -2 & -1 \\ -1 & -2 & -3 & -3 & -3 & -2 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ +1 & +2 & +3 & +3 & +3 & +2 & +1 \\ +1 & +2 & +2 & +2 & +2 & +2 & +1 \\ +1 & +1 & +1 & +1 & +1 & +1 & +1 \end{bmatrix} \quad (34)$$

На овај начин се пикселима који су удаљенији од централног пиксела додељују мање тежине у односу на пикселе који су ближе централном пикселу.

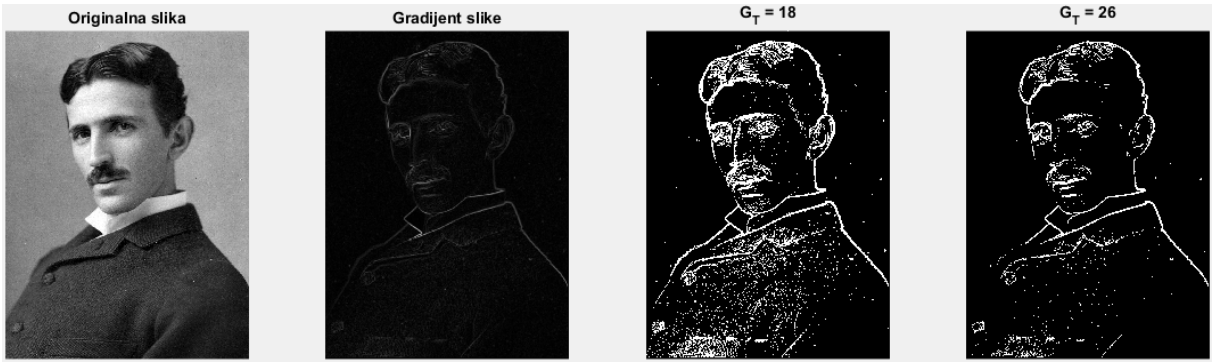
На слици 55 је дат приказ начина имплементације Абдуовог оператора у Матлабу.

```
Hx = [1 1 1 0 -1 -1 -1;
      1 2 2 0 -2 -2 -1;
      1 2 3 0 -3 -2 -1;
      1 2 3 0 -3 -2 -1;
      1 2 3 0 -3 -2 -1;
      1 2 2 0 -2 -2 -1;
      1 1 1 0 -1 -1 -1];
Hy = [-1 -1 -1 -1 -1 -1 -1;
      -1 -2 -2 -2 -2 -2 -1;
      -1 -2 -3 -3 -3 -2 -1;
       0 0 0 0 0 0 0;
       1 2 3 3 3 2 1;
       1 2 2 2 2 2 1;
       1 1 1 1 1 1 1];

for i=1:size(inputImage, 1)-6
    for j=1:size(inputImage, 2)-6
        Gx = sum(sum(Hx.*inputImage(i:i+6, j:j+6)))/34;
        Gy = sum(sum(Hy.*inputImage(i:i+6, j:j+6)))/34;
        gradient(i,j) = sqrt(Gx.^2 + Gy.^2);
    end
end
```

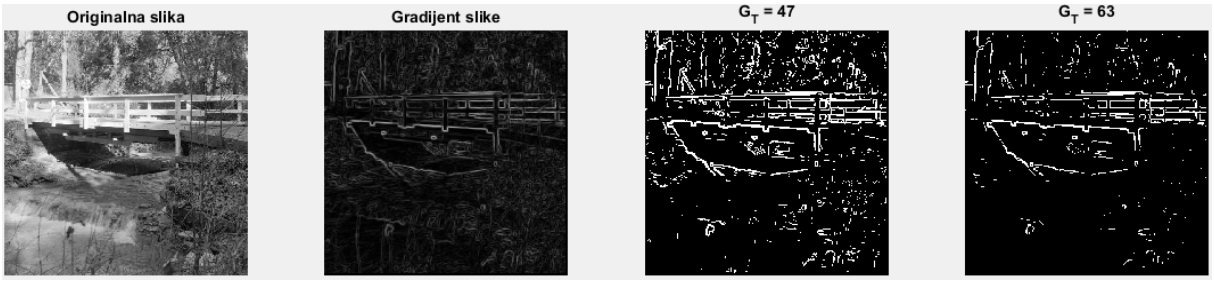
Слика 55. Имплементација Абдуовог оператора у Матлабу

На сликама испод (56-60) је приказан рад овог оператора на сликама различитих карактеристика и постављеном вредношћу прага G_T тако да се 10% и 5% пиксела са највећим градијентом прогласи ивицама.

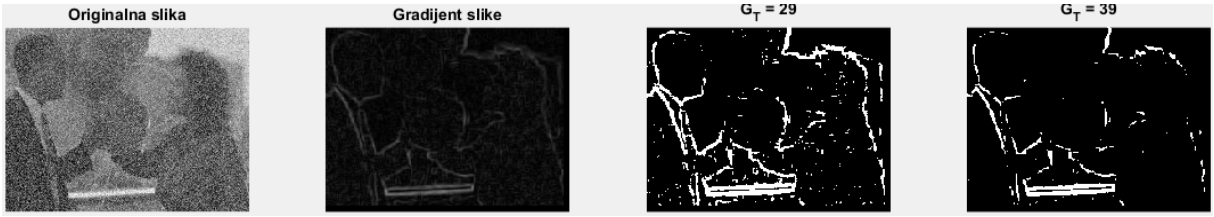
Слика 56. Приказ рада Абдуовог оператора на слици ивица оштрог интензитета



Слика 57. Приказ рада Абдуовог оператора на слици са шумом



Слика 58. Приказ рада Абдуовог оператора на слици са доста детаља



Слика 59. Приказ рада Абдуовог оператора на слици са доста шума

Слика 60. Приказ рада Абдуовог оператора на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, Абдуов оператор даје веома добре резултате на слици оштрих ивица, Лошије резултате даје на слици ивица слабијег интензитета и слици са доста детаља. У поређењу са претходним операторима, боље се показује и на слици са доста шума, детектује мање лажних ивица.

6.2.8. DroG оператор

Додатно боље резултате дају оператори код којих се за процес усредњавања користи тежинска функција Гаусовог облика. Нека су:

$$g(x,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{x^2}{2\sigma^2}} \tag{35}$$

$$g(y,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{y^2}{2\sigma^2}} \tag{36}$$

континуалне Гаусове функције са стандардном девијацијом σ.

Сви наведени градијентни оператори, осим оператора разлике, представљају комбинацију операција усредњавања и формирања разлике. Здружени импулсни одзив оператора се може представити конволуцијом:

$$h_x[m,n] = h_{gx}[m,n] * h_{NF}[-m,-n] \tag{37}$$

где је $h_{gx}[m,n]$ неки од градијентних оператора, а $h_{NF}[-m,-n]$ импулсни одзив нискофреквентног филтра за усредњавање.

Један од примера здруженог оператора је DroG оператор који представља комбинацију нискофреквентног Гаусовог филтра и диференцијатора. Импулсни одзиви оператора се добијају диференцирањем дводимензионалне Гаусове функције:

$$h_x(x,y) = \frac{\partial}{\partial x}[g(x,\sigma)g(y,\sigma)] = -\frac{xg(x,\sigma)g(y,\sigma)}{\sigma^2} = -\frac{x}{\sqrt{2\pi}\sigma^3} e^{\frac{x^2+y^2}{2\sigma^2}} \tag{38}$$

$$h_y(x,y) = \frac{\partial}{\partial y}[g(x,\sigma)g(y,\sigma)] = -\frac{yg(x,\sigma)g(y,\sigma)}{\sigma^2} = -\frac{y}{\sqrt{2\pi}\sigma^3} e^{\frac{x^2+y^2}{2\sigma^2}} \tag{39}$$

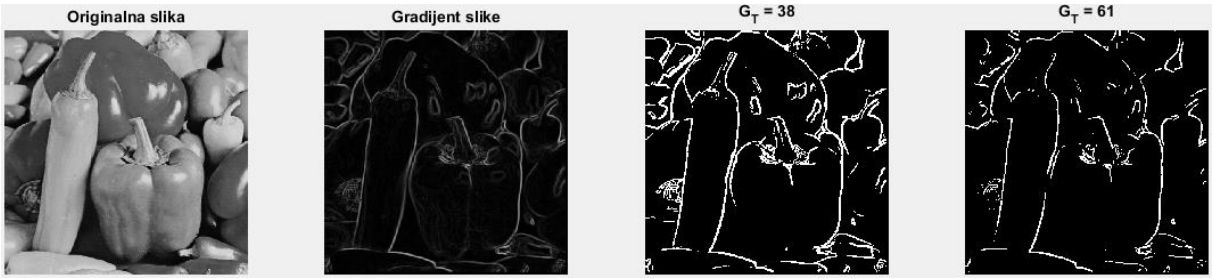
Параметар σ Гаусове функције служи за контролисање процеса усредњавања. Повећањем овог параметра се смањује утицај шума и поправља квалитет детектора ивица. Међутим, сувише велике вредности овог параметра изазивају ефекте заобљавања углова и погоршање локализације ивица. Због тога је избор оптималне вредности σ резултат компромиса између добре детекције и добре локализације ивице [2].

На слици 61 је дат приказ начина имплементације DroG оператора у Матлабу.

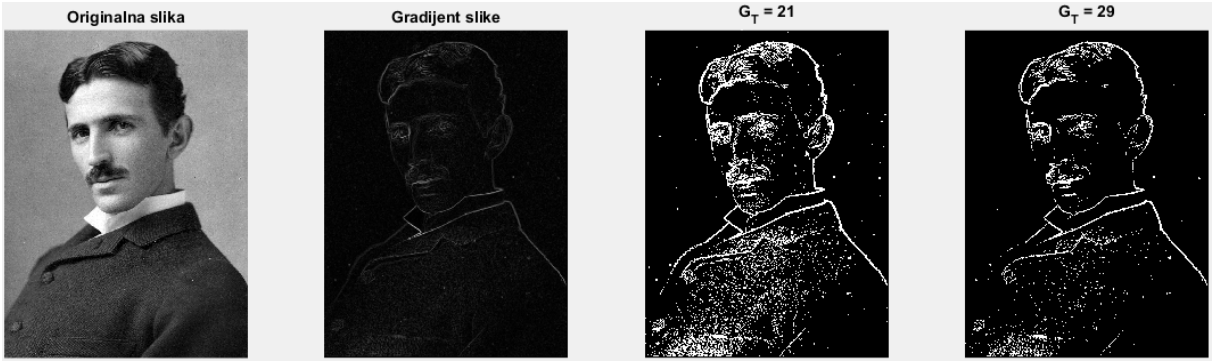
```
sigma = 2.0;
kernelSize = 5;

lin = round(linspace(-floor(kernelSize/2),floor(kernelSize/2),kernelSize));
[X,Y] = meshgrid(lin,lin);
hg = exp((X.^2 + Y.^2)/(2*(sigma^2)));

Hx_temp = (-X./(sqrt(2*pi)*sigma^3).*hg);
Hy_temp = (-Y./(sqrt(2*pi)*sigma^3)).*hg;

Hx = Hx_temp - sum(Hx_temp(:))/kernelSize^2;
Hy = Hy_temp - sum(Hy_temp(:))/kernelSize^2;

for i=1:size(inputImage, 1)-kernelSize+1
    for j=1:size(inputImage, 2)-kernelSize+1
        Gx = sum(sum(Hx.*inputImage(i:i+kernelSize-1, j:j+kernelSize-1)))...
            /sum(sum(max(Hx, 0), 2));
        Gy = sum(sum(Hy.*inputImage(i:i+kernelSize-1, j:j+kernelSize-1)))...
            /sum(sum(max(Hy, 0), 2));

        gradient(i,j) = sqrt(Gx.^2 + Gy.^2);
    end
end
```

Слика 61. Имплементација DroG оператора у Матлабу

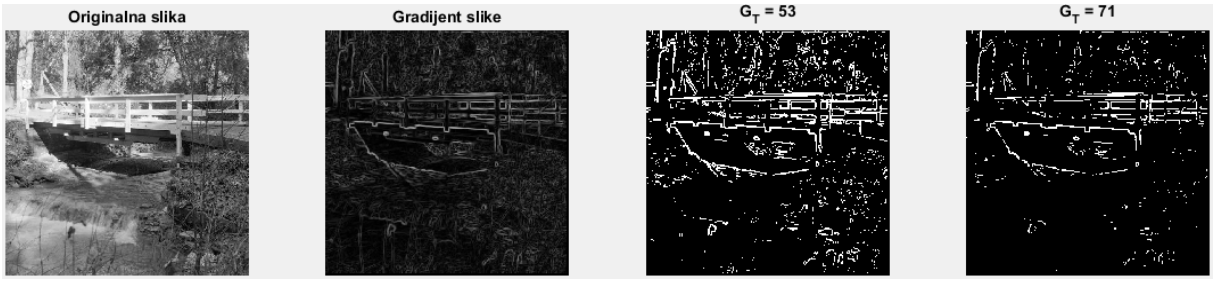
На сликама испод (62-66) је приказан рад овог оператора на сликама различитих карактеристика и постављеном вредношћу прага G_T тако да се 10% и 5% пиксела са највећим градијентом прогласи ивицама.



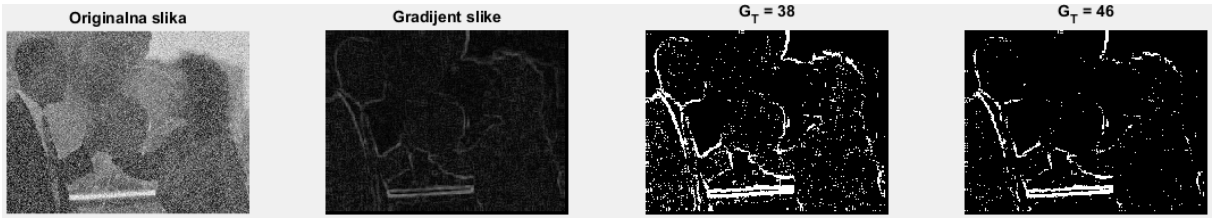
Слика 62. Приказ рада DroG оператора на слици ивица оштрог интензитета



Слика 63. Приказ рада DroG оператора на слици са шумом



Слика 64. Приказ рада DroG оператора на слици са доста детаља



Слика 65. Приказ рада DroG оператора на слици са доста шума

Слика 66. Приказ рада DroG на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, DroG оператор даје добре резултате на слици са оштрим ивицама. Задовољавајуће резултате бележи на слици слабијег интензитета и слици са доста детаља. На сликама са шумом даје сличне резултате као и Абдуов оператор.

6.2.9. Канијев оператор

Сви до сада поменути оператори су добијени на хеуристички начин, док се до Канијевог оператора долази аналитичким путем. У извођењу Канијевог оператора се полази од једнодимензионалне континуалне идеално стрме ивице, амплитуде h_g , којој је суперпониран бели Гаусов шум стандардне девијације σ_n. Издвајање ивица се врши конволуцијом у континуалном домену између сигнала ивице са шумом $f(x)$ и антисиметричног импулсног одзива оператора $h(x)$ који има ненулту вредност само у интервалу $[-w, w]$. Положај ивице је одређен локалним максимумом конволуције $f(x) * h(x)$ [2]. Кани је формулисао три услова која би идеални оператор требало да испуњава:

- Мали проценат лажних ивица, односно висок проценат правих ивица
- Добра локализација
- Јединични одзив на ивицу (све ивице су ширине 1 пиксел)

Добра детекција ивица се може постићи максимизацијом амплитуде односа сигнал/шум градијента. Тиме се добија мала вероватноћа грешке пропуштања правих ивичних тачака

и мала вероватноћа лажног означавања тачака које нису ивице. Однос сигнал/шум код коришћеног модела ивице се може записати у облику:

$$SNR = \frac{h_E}{\sigma_n} S(h) = \frac{h_E}{\sigma_n} \frac{\int_{-w}^{0} h(x)\,dx}{\sqrt{\int_{-w}^{w} [h(x)]^2\,dx}} \tag{40}$$

где h_E представља амплитуду идеално стрме ивице, σ_n стандардну девијацију Гаусовог шума, а $h(x)$ функција оператора.

Добра локализација ивице се добија ако ивичне тачке које се добијају применом оператора леже што је могуће ближе положају идеалне ивице. Фактор локализације се дефинише као:

$$LOC = \frac{h_E}{\sigma_n} L(h) = \frac{h_E}{\sigma_n} \frac{|h'(0)|}{\sqrt{\int_{-w}^{w} [h'(x)]^2\,dx}} \tag{41}$$

Ширина ивице од једног пиксела је веома битна за даљи процес анализе слике, јер се у противном мора применити процес стањивања ивице. Нека је растојање максимума градијента када је у слици присутан шум, x_m, пропорционално ширини интервала w, тј:

$$x_m = kw \tag{42}$$

Канијев оптимизациони поступак се састоји у максимизацији производа $S(h)L(h)$ под ограничењем (42). Због сложености оптимизационог поступка не може се добити аналитичко решење проблема, али се може применити поступак нумеричке оптимизације. За мале вредности параметра $x_m < 0{,}5$, импулсни одзив Канијевог оператора $h(x)$ је сличан импулсном одзиву бокскар оператора. За веће вредности параметра $x_m > 1{,}2$, импулсни одзив Канијевог оператора је сличан импулсном одзиву DroG оператора. Идентичним поступком се долази и до оператора за издвајање вертикалне компоненте градијента.

У дводимензионалном случају оптимални детектор мора задовољити услов:

$$\frac{\partial^2}{\partial n^2} [g(x, y, \sigma) * f(x, y)] = 0 \tag{43}$$

где $g(x, y, \sigma)$ представља импулсни одзив Гаусовог филтра, а n јединични вектор у смеру градијента дефинисан као:

$$n = \frac{G(x,y)}{|G(x,y)|} \tag{44}$$

Диференцирање у правцу градијента је у ствари диференцирање у правцу који је нормалан на правац простирања ивице. Због тога нула другог извода (43) одговара превојној тачки ивице, односно максимуму градијента. Због тога је прва фаза Канијевог оператора у ствари идентична примени DroG оператора на слику.

После одређивања градијентне слике, врши се елиминација тачака код којих градијент не задовољава услов (43) (потискивање не-максимума). Потискивање не-максимума врши се на врло једноставан начин. Пошто су за сваки пиксел слике познати модуо $G(m, n)$ и смер градијента $\theta_G[m, n]$, израчунати модуо градијента се пореди са вредностима градијента у две тачке које леже еквидистантно са обе стране ивице на правцу градијента за посматрани пиксел. Вредност градијента се задржава само ако је већа од обе вредности са којима се пореди, у противном се изједначава са нулом. На тај начин се апроксимативно одређује максимум првог извода у правцу нормалном на правац ивице. После потискивања не-максимума треба применити уобичајени поступак поређења са прагом ради формирања бинарне мапе ивица.

Дискретизацијом континуалних импулсних одзива оператора добијају се импулсни одзиви за издвајање ивица из дигиталних слика. Као и код DroG оператора, ширина прозора мора бити довољно велика (око 8σ) да би се спречила појава преклапања у фреквенцијском домену због коначних димензија прозора. Због тога, као и због додатног поступка потискивања не-максимума, Канијев оператор има мању рачунску ефикасност од претходно описаних градијентних оператора, па је код његове примене потребно обратити велику пажњу на минимизацију броја рачунских операција.

Иако је рачунски комплексан, Канијев оператор издвајања ивица има низ предности над класичним операторима. Пре свега, овај оператор обезбеђује најбољу детекцију ивица, као и могућност компромиса између детекције и локализације у зависности од односа сигнал/шум, а који се остварује променом вредности параметра σ. Осим тога, у генерисаној мапи ивица, све ивице имају ширину од једног пиксела, што значи да није потребно вршити истањивање ивица које је неопходно ако се примењују други оператори.

Укратко описано, Канијев оператор се састоји од примене пет корака:

1. Примена Гаусовог филтра ради уклањања шума у слици
2. Рачунање интензитета градијента слике применом неког од оператора (најчешће Собел)
3. Рачунање магнитуде и угла градијента
4. Потискивање не-максимума
5. Поступак поређења са дефинисаним прагом

На слици 67 је дат приказ начина имплементације потискивања не-максимума код Канијевог оператора у Матлабу.

```
% Non-Maximum Supression
for i=2:rows-1
    for j=2:columns-1
        if (arg2(i,j)==0)
            BW(i,j) = (magnitude(i,j) == max([magnitude(i,j),...
                magnitude(i,j+1), magnitude(i,j-1)]));
        elseif (arg2(i,j)==45)
            BW(i,j) = (magnitude(i,j) == max([magnitude(i,j),...
                magnitude(i+1,j-1), magnitude(i-1,j+1)]));
        elseif (arg2(i,j)==90)
            BW(i,j) = (magnitude(i,j) == max([magnitude(i,j)...
                , magnitude(i+1,j), magnitude(i-1,j)]));
        elseif (arg2(i,j)==135)
            BW(i,j) = (magnitude(i,j) == max([magnitude(i,j),...
                magnitude(i+1,j+1), magnitude(i-1,j-1)]));
        end
    end
end

BW = BW.*magnitude;
```

Слика 67. Потискивање не-максимума

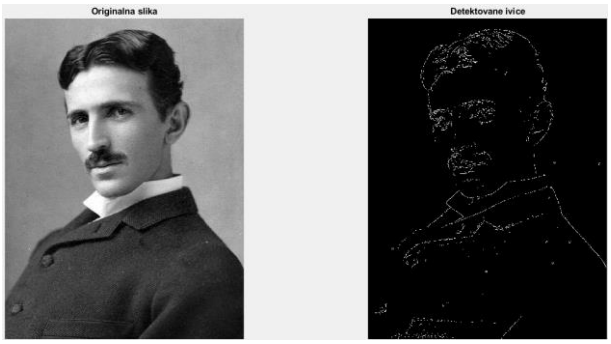
На слици 68 је приказан код за поређење са дефинисаним прагом код Канијевог оператора у Матлабу.

```
% Hysteresis Thresholding
tLow = tLow * max(max(BW));
tHigh = tHigh * max(max(BW));
T_res = zeros(rows, columns);
for i = 1  : rows -1
    for j = 1 : columns -1
        if (BW(i, j) < tLow)
            T_res(i, j) = 0;
        elseif (BW(i, j) > tHigh)
            T_res(i, j) = 1;
        % 8-connected components
        elseif ( BW(i+1,j)>tHigh || BW(i-1,j)>tHigh || BW(i,j+1)>tHigh ...
                || BW(i,j-1)>tHigh || BW(i-1, j-1)>tHigh ||...
                BW(i-1, j+1)>tHigh || BW(i+1, j+1)>tHigh || BW(i+1, j-1)>tHigh)
            T_res(i,j) = 1;
        end
    end
end
edge_final = uint8(T_res.*255);
```

Слика 68. Поређење са дефинисаним прагом код Канијевог оператора

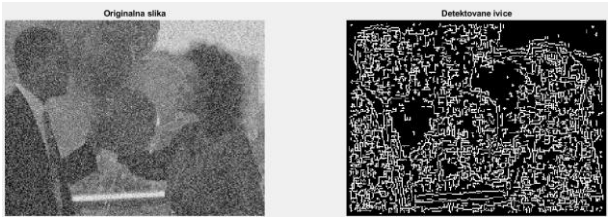
На сликама испод (69-73) ће бити приказан рад овог оператора на сликама различитих карактеристика.

Слика 69. Приказ рада Канијевог оператора на слици ивица оштрог интензитета

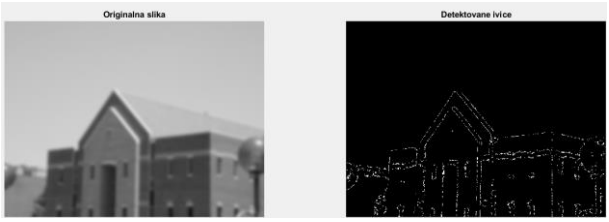


Слика 70. Приказ рада Канијевог оператора на слици са шумом

Слика 71. Приказ рада Канијевог оператора на слици са доста детаља



Слика 72. Приказ рада Канијевог оператора на слици са доста шума



Слика 73. Приказ рада Канијевог на слици са ивицама слабијег интензитета

Канијев оператор бележи одличне резултате на слици са оштрим ивицама. Добре резултате даје на слици са ивицама слабијег интензитета и слици са доста детаља. На слици са мање шума даје значајно боље резултате у поређењу са претходним операторима.

6.3. Лапласови оператори

Један од недостатака описаних метода за издвајање ивица слике је то што дају лоше резултате када прелаз са једног на други ниво сивог није оштар. У наведеном случају је градијент слике мали, па се положај ивице не може се поуздано одредити у процесу формирања мапе ивица [2]. Једно решење овог проблема је употреба другог извода слике уместо првог као код градијентних оператора:

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} \tag{45}$$

Најпростији облик дискретне апроксимације једначине (45) је:

$$G[m,n] = \{(f[m,n] - f[m,n-1]) - (f[m,n+1] - f[m,n])\} + \{(f[m,n] - f[m+1,n]) - (f[m-1,n] - f[m,n])\} = 4f[m,n] - f[m,n-1] - f[m,n+1] - f[m-1,n] \tag{46}$$

6.3.1. Лапласов оператор

За примену Лапласовог оператора једначина (45) се може изразити преко збира две конволуционе маске:

$$H = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & -1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ 0 & -2 & 0 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{47}$$

који се после нормализације збира позитивних (или негативних) пиксела на јединицу своди на коначни облик:

$$H = \frac{1}{4}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{48}$$

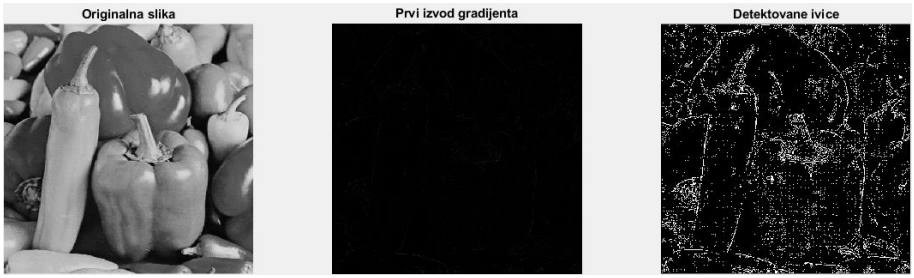
Међутим примена Лапласовог параметра за издвајање ивица има и недостатке. Осетљивост на шум је доста већа у односу на градијентне операторе због коришћења другог извода слике. Такође, губи се информација о смеру ивице због скаларне природе једначине (45). Ако се после примене овог оператора примени поређење са прагом, у мапи ивица ће се појавити двоструке ивице [2].

На слици 74 је дат приказ начина имплементације Лапласовог оператора у Матлабу.

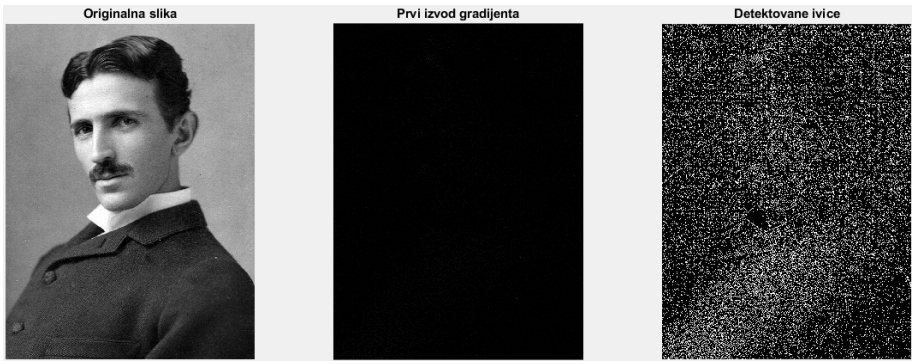
```
H = [0 -1 0; -1 4 -1; 0 -1 0];

for i=1:size(inputImage, 1)-2
    for j=1:size(inputImage, 2)-2
        gradientFirstDerivative(i+1,j+1) = sum(sum(H.*inputImage(i:i+2, j:j+2)))/4;
    end
end
```

Слика 74. Имплементација Лапласовог оператора у Матлабу

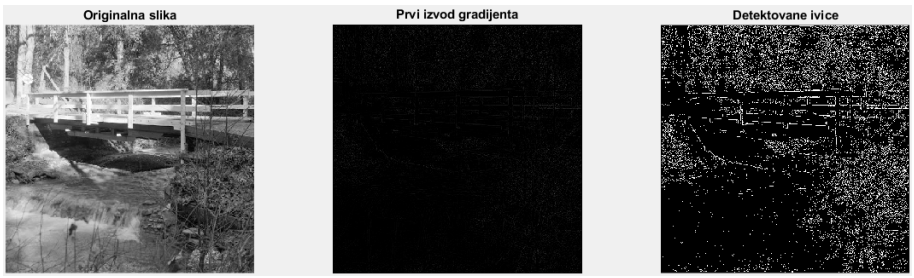
На сликама испод (75-79) ће бити приказан рад овог оператора на сликама различитих карактеристика.



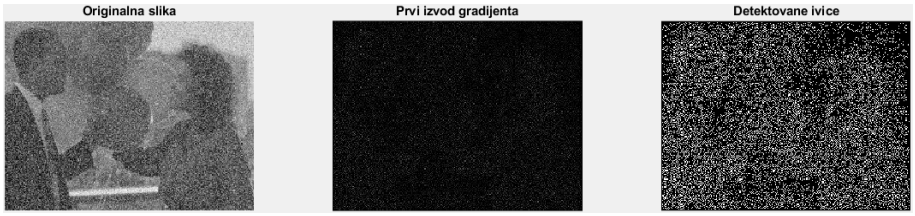
Слика 75. Приказ рада Лапласовог оператора на слици ивица оштрог интензитета



Слика 76. Приказ рада Лапласовог оператора на слици са шумом



Слика 77. Приказ рада Лапласовог оператора на слици са доста детаља



Слика 78. Приказ рада Лапласовог оператора на слици са доста шума

Слика 79. Приказ рада Лапласовог на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, Лапласов оператор даје доста лошије резултате на свим сликама у поређењу са претходно описаним побољшаним градијентим операторима. Најбољи резултати су на слици са оштрим ивицама и слици са доста детаља, доста лошији резултати на слици са ивицама слабијег интензитета, док су резултати добијени на сликама са шумом готово неупотребљиви.

6.3.2. Лаплас-Гаусов оператор

Претходно описани Лапласов оператор се ретко користи у пракси управо због велике осетљивости на шум. Међутим, уколико се пре примене Лапласовог оператора из слике уклони шум, он даје добре резултате. Шум из слике се може уклонити филтрирањем помоћу нискофреквентног филтра, чији је импулсни одзив сепарабилан и има облик Гаусове криве са стандардном девијацијом σ [2]. Импулсни одзив континуалног дводимензионалног Гаусовог филтра је:

$$h_{NF}(x,y) = \left[\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}\right]\left[\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{y^2}{2\sigma^2}}\right] \tag{49}$$

Због тога што су и филтрирање и апроксимација другог извода линеарне операције, могуће је дефинисати здружени импулсивни одзив за обе операције, који се назива Лаплас-Гаусов оператор или ЛоГ оператор и представља Лапласијан импулсног одзива:

$$h_{LoG}(x,y) = \frac{1}{\pi\sigma^4}\left(1 - \frac{x^2+y^2}{2\sigma^2}\right) e^{\frac{x^2+y^2}{2\sigma^2}} \tag{50}$$

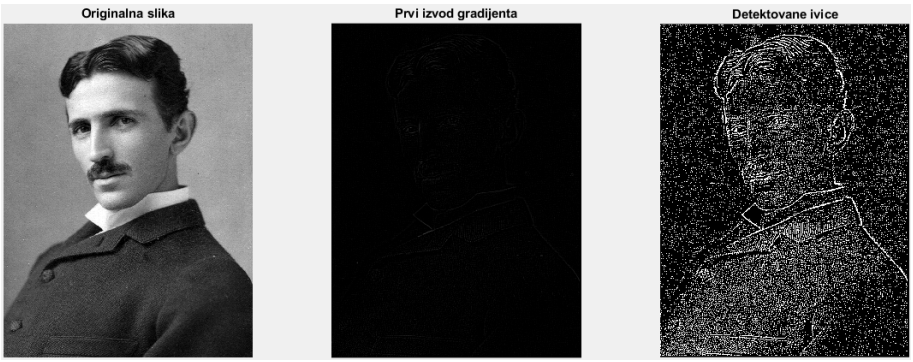
На слици 80 је дат приказ начина имплементације ЛоГ оператора у Матлабу.

```
kernelSize = 7;
sigma = 1.7;

lin = round(linspace(-floor(kernelSize/2),floor(kernelSize/2),kernelSize));
[X,Y] = meshgrid(lin,lin);
hg = exp((X.^2 + Y.^2)/(2*(sigma^2)));

H_temp = hg.*(1/(pi*sigma^4))*(1-(X.^2 + Y.^2)/(2*(sigma^2)));
H = H_temp - sum(H_temp(:))/kernelSize^2;

for i=1:size(inputImage, 1)-kernelSize+1
    for j=1:size(inputImage, 2)-kernelSize+1
        gradientFirstDerivative(i+1,j+1) = ...
            sum(sum(H.*inputImage(i:i+kernelSize-1, j:j+kernelSize-1)))...
            /sum(sum(max(H, 0), 2));
    end
end
```

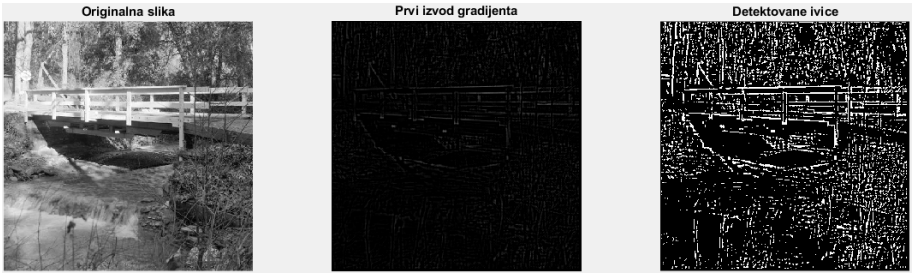
Слика 80. Имплементација ЛоГ оператора у Матлабу

На сликама испод (81-85) ће бити приказан рад овог оператора на сликама различитих карактеристика.

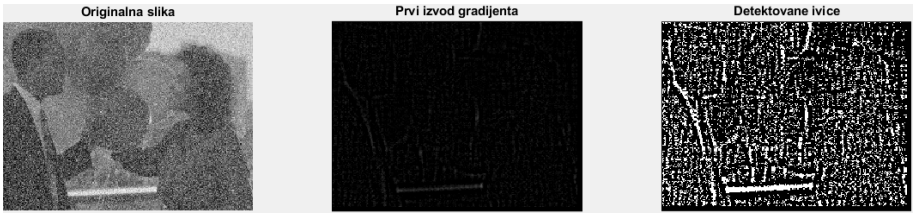
Слика 81. Приказ рада ЛоГ оператора на слици ивица оштрог интензитета



Слика 82. Приказ рада ЛоГ оператора на слици са шумом



Слика 83. Приказ рада ЛоГ оператора на слици са доста детаља



Слика 84. Приказ рада ЛоГ оператора на слици са доста шума

Слика 85. Приказ рада ЛоГ на слици са ивицама слабијег интензитета

Као што се може видети на претходним сликама, ЛоГ оператор даје добре резултате на слици са оштрим ивицама. Задовољавајући резултати су добијени на слици са доста детаља и слици са ивицама слабог интензитета, док су лошији резултати на сликама са шумом, али у поређењу са претходно описаним Лапласовим оператором, резултати на слици са мање шума су много бољи.

7. Сегментација дигиталне слике

Сегментација је корак у обради дигиталне слике који често следи након корака издвајања ивица. Термин сегментација се односи на групу поступака за поделу слике на регионе са сличним атрибутима. Од атрибута за поређење се код монохромних слика најчешће користи осветљеност, али се поред осветљености могу користити и обележја попут ивица и текстура. Основна подела техника сегментације је на технике које се заснивају на проналажењу ивица – издвајају само оне пикселе који припадају ивицама објеката, и технике које се заснивају на издвајању региона – издвајају цео објекат од позадине придружујући пикселе чија је осветљеност испод дефинисаног прага позадини, а остале објекту.

Постоји доста техника сегментације које се могу пронаћи у литератури, а најчешће коришћене су [2]:

- Сегментација помоћу прага - група метода које се заснивају на поређењу осветљености пиксела са једним или више дефинисаних прагова
    - Сегментација са једним прагом
    - Сегментација са више прагова
- Сегментација помоћу кластеризације - процес поделе према сличности вектора података на основу дефинисаних карактеристика
- Сегментација помоћу региона - обухвата методе које користе обележја израчуната на просторном суседству пиксела
    - Сегментација помоћу раста региона

    - Сегментација помоћу раздвајања и спајања региона
- Сегментација помоћу граница региона – сегментација на основу претходно издвојених ивица операторима за издвајање ивица у слици
    - Спајање ивица фитовањем криве
    - Спајање ивица хеуристичким методама
    - Спајање ивица Хафовом трансформацијом
- Сегментација текстура – базира се на израчунавању неке мере текстуре за сваки пиксел слике

У даљем делу рада ће детаљније бити описан процес сегментације заснован на издвајању ивица.

7.1. Сегментација помоћу граница региона

Сегментација помоћу граница региона обухвата два корака:

- Издвајање ивица у слици (детектовање ивичних пиксела) коришћењем неког од претходно описаних оператора за издвајање ивица
- Повезивање ивица – подразумева повезивање ивичних пиксела у ивице

7.1.1. Поступак повезивања ивица

У идеалном случају, детекција ивица би требало да издвоји само сет пиксела који леже на ивицама региона. У пракси то није случај, већ се поред пиксела који стварно леже на ивицама објеката јављају и пиксели који су последица присуства шума у слици. Из тог разлога је детекција ивица често праћена поступком повезивања ивица чији је циљ повезивање граничних пиксела у смислене ивице и/или границе региона.

7.1.1.1. Локално процесирање

Једноставан приступ повезивања ивичних тачака је анализа карактеристика пиксела блиског суседства (најчешће 3×3 или 5×5) око сваке тачке (x, y), која је декларисана као ивична тачка на основу претходно примењене технике издвајања ивица. Све тачке које су сличне на основу дефинисаних критеријума се повезују са тачком (x, y), формирајући ивицу пиксела који деле заједничке особине на основу дефинисаног критеријума. Две главне особине које се користе за дефинисање сличности ивичних пиксела у оквиру локалног процесирања су јачина (магнитуда) и смер вектора градијента. Ивични пиксел (x', y') који се налази у суседству пиксела (x, y) је сличан пикселу (x, y) по критеријуму магнитуде ако је:

$$|\nabla f(x, y) - \nabla f(x', y')| \leq T \tag{51}$$

где је T дефинисана вредност прага.

Ивични пиксел (x', y') који се налази у суседству пиксела (x, y) је сличан пикселу (x, y) по критеријуму угла ако је:

$$|\alpha(x, y) - \alpha(x', y')| \leq A \tag{52}$$

где је A дефинисани праг угла [14].

Пиксел чије су координате (s, t) и који припада сету S_{xy} – сету координата суседа централног пиксела (x, y), ће се узети у обзир за повезивање са пикселом (x, y) ако задовољава претходно описане критеријуме магнитуде и смера градијента. Овај процес се понавља за сваки ивични пиксел. Како се центар суседства помера од пиксела до пиксела, запис повезаних тачака се чува. Једноставан начин процедуре чувања је додељивање различите вредности интензитета сваком сету повезаних ивичних пиксела.

Оваква процедура је рачунарски веома захтевна јер је потребно испитати све суседе сваке тачке. Међутим, она се може поједноставити следећим корацима [3]:

1. Израчунати магнитуду градијента и низове углова, $M(x, y)$ и $\alpha(x, y)$ улазне слике $f(x, y)$
2. Формирати бинарну слику, $g(x, y)$ чија је вредност у било којој тачки (x, y) једнака:

$$g(x, y) = \begin{cases} 1 & \text{ако је } M(x, y) > T_M \text{ и } \alpha(x, y) = A \pm T_A \\ 0, & \text{у супротном} \end{cases} \tag{53}$$

   где је T_M вредност прага, A дефинисани смер угла, а $\pm T_A$ дефинише скуп прихватљивих смерова за A

3. Скенирати редове функције g и попунити (поставити на 1) све празнине (сетове нула) у сваком реду које не прелазе наведену дужину L.
4. За детектовање празнина у било ком другом смеру, θ, потребно је ротирати g за овај угао и применити процедуру хоризонталног скенирања из корака 3. Након тога је потребно ротирати резултат назад за - θ.

7.1.1.2. Глобално процесирање

Локално процесирање је применљиво у ситуацијама када постоје информације о пикселима који припадају индивидуалним објектима. Међутим, чест је случај да ове информације нису доступне и да је потребно радити са неструктуираним окружењима у којима постоји само мапа ивица без информација о томе где се налази објекат од интереса. У описаном случају, сви пиксели су кандидати за повезивање, и као такви морају бити прихваћени или одбачени на основу предефинисаних особина.

Код глобалног процесирања ивичне тачке се повезују на основу тога да ли леже на истој кривој одређеног облика. Након детекције, ове криве формирају ивице или границе региона од интереса. За разлику од локалног процесирања, код глобалног процесирања се узимају у обзир глобални односи између пиксела.

Претпоставимо да је за n тачака слике потребно пронаћи подсетове ових тачака који леже на правим линијама. Једно решење јесте пронаћи све линије одређене сваким паром ових тачака, након тога пронаћи све подсетове тачака које су близу одређених линија. Овакав

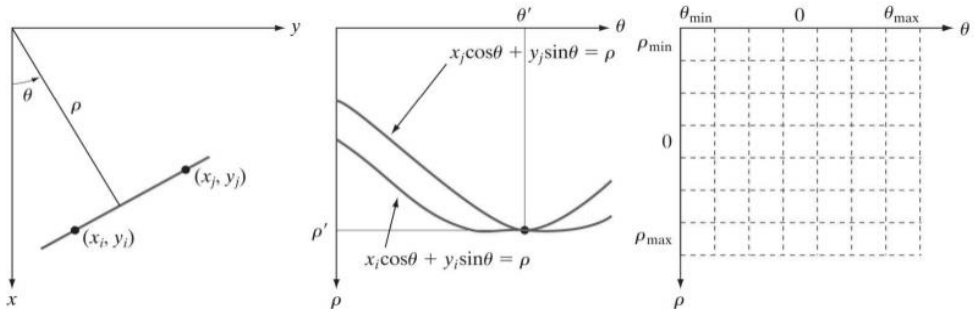
поступак подразумева проналажење $\frac{n(n-1)}{2} \approx n^2$ линија и предузимање $\frac{n\cdot(n(n-1))}{2} \approx n^3$ поређења сваке тачке са сваком линијом што представља веома захтеван процес [3].

7.1.1.2.1. Хафова трансформација

Хаф (енг. *Hough*) је предложио алтернативно решење претходно описаног проблема глобалног процесирања које се назива Хафова трансформација. Хафова трансформација подразумева пресликавање линије из правоуглог координатног система у тачку у поларном координатном систему, па се права линија може представити у параметарском облику:

$$\rho = x\cos\theta + y\sin\theta \tag{54}$$

где је ρ растојање од координатног почетка до праве линије дуж нормале на линију, а θ угао између нормале и x-осе. Тачка са координатама (ρ', θ') представља линију која пролази кроз обе тачке у xy-равни. На слици 86 је графички приказан процес пресликавања линије у тачку у поларном домену [3].



Слика 86. Пресликавање праве у поларни координатни систем

На слици 87 је дат приказ Матлаб кода за пресликавања ивичних пиксела слике у Хафов простор и креирање акумулатора.

```
[width,height] = size(outputImage);
rhoLimit = ceil(norm([width-1 height-1]));
rho = (-rhoLimit:1:rhoLimit);
theta = (-90:1:89);
[x,y,val] = find(outputImage);
h = zeros(length(rho), length(theta));

for k=1:length(val)
    for t=1:length(theta)
        r = (x(k,1)-1)*cos(theta(1,t)*pi/180) + ...
            (y(k,1)-1)*sin(theta(1,t)*pi/180);
        rho_index = find(rho==round(r));
        h(rho_index, t) = h(rho_index, t) + 1;
    end
end
```

Слика 87. Имплементација Хафове трансформације у Матлабу

Спајање ивица помоћу Хафове трансформације

Хафова трансформација се може користити за повезивање ивичних тачака на основу доношења одлуке (гласања) о томе да ли леже на истој кривој дефинисаног облика.

Дуда и Харт су описали прву примену Хафове трансформације за спајање ивица у бинарној слици. Свака тачка (x_i, y_i) из xy-равни чија амплитуда није нула се пресликава у синусоиду у $\rho\theta$-равни. Ако се параметри ρ и θ квантују, онда се $\rho\theta$-раван може поделити на ћелије, па се после сваког пресликавања пиксела инкрементирају све оне ћелије кроз које пролази добијена крива. После пресликавања свих пиксела, врши се испитивање стања свих ρ, θ ћелија акумулатора. Свака ћелија са високим садржајем одговара пикселима слике који леже на истој правој са параметрима ρ, θ. Колинеарна тачка M која лежи на линији $x_i \cos\theta + y_i \sin\theta = \rho$ даје M кривих које се секу у тачки (ρ_i, θ_i) у поларном простору.

Као што је већ речено, свака (ρ, θ) линија чија је амплитуда довољно велика представља једну линију у оригиналној слици. Ако се таква линија преклопи са мапом ивица, лако се могу открити недостајуће ивичне тачке, или чак цели ивични сегменти. Користећи нека хеуристичка правила, мапа ивица се може допунити недостајућим ивичним пикселима. Други начин за допуњавање мапе ивица јесте да се Хафова трансформација комбинује са морфолошким операцијама изведеним над прозорима малих димензија [2].

Алгоритам за повезивање ивица помоћу Хафове трансформације се састоји из следећих корака:

1. Примена неког од оператора за детекцију ивице на оригиналној слици
2. Одређивање опсега ρ и θ параметара - најчешће је опсег параметра θ постављен на $[-90,90]$ степени, а ρ на $[-d, d]$, где је d дужина дијагонале слике
3. Креирање дводимензионалног низа који се назива акумулатор и који представља Хафов простор димензија (број ρ-подеока, број θ-подеока) и постављање свих вредности овог низа на 0
4. За сваки пиксел слике издвојених ивица, проверити да ли је пиксел ивични пиксел; уколико јесте ивични пиксел, проћи кроз све могуће вредности θ, израчунати одговарајућу вредност ρ, пронаћи θ и ρ индексе акумулатора и инкрементирати акумулатор на основу вредности тих парова
5. Испитивање вредности акумулатора и издвајање оних ћелија које имају велику концентрацију пиксела; ако је вредност већа од дефинисане вредности прага, узети θ и ρ индекс и вредности θ и ρ из пара индекса и конвертовати их у линију у правоуглом координатном систему ако су испуњени услови за повезивање пиксела

Испитивање везе између пиксела издвојене ћелије, односно услова за повезивање пиксела подразумева испитивање удаљености између пиксела и спајање истих уколико је растојање мање од унапред одређеног прага.

На слици 88 је приказан начин спајања ивичних сегмената помоћу Хафове трансформације.

Слика 88. Спајање ивица употребом Хафове транформације

Хафова трансформација је концептуално једноставна и лака за имплементацију и може се користити и за друге облике, а не само за линије. Међутим, Хафова трансформација је комплексна за објекте који имају доста параметара.

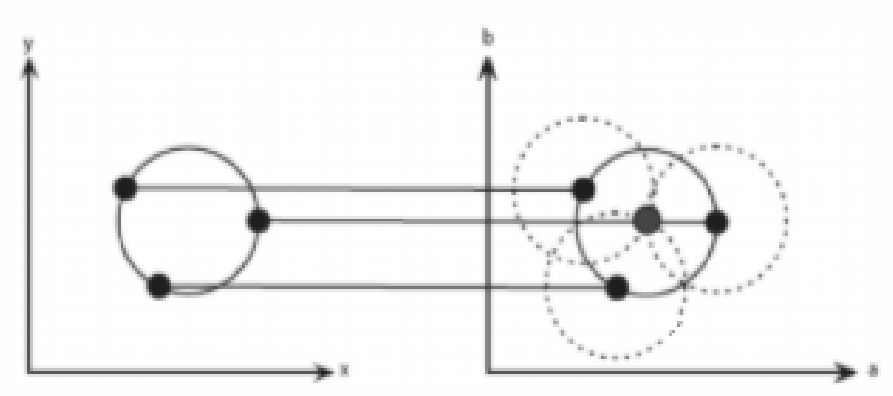
Спајање ивица помоћу Хафове трансформације за кружнице

Претходно описана формулација Хафове трансформације се користи за повезивање пиксела правим линијама. За повезивање пиксела кривим, односно кружним линијама, се користи Хафова трансформација за кружнице.

Хафова трансформација за кружнице је модификована верзија стандардне Хафове трансформације. Једначина кружнице је дата формулом:

$$(x-a)^2 + (y-b)^2 = r^2 \tag{55}$$

где (x, y) представља координате пиксела на слици, (a, b) координате центра кружнице, а r полупречник кружнице. Свака тачка (x, y) се у параметарском простору трансформише у скуп кружница са средиштем у координатама (x, y). Уколико је познат полупречник почетне кружнице, свака тачка те кружнице се трансформише у нову кружницу. Када се овај поступак примени над свим тачкама кружнице, добија се скуп кружница истог полупречника које се секу у једној тачки чије координате одређују координате центра почетне кружнице (слика 89) [15].



Слика 89. Трансформација тачака кружнице у параметарски простор

У случају када полупречник кружнице није познат, поступак се понавља за вредности полупречника $r = 1,2,3, \dots, n$, где је n дужина дијагонале слике.

Поступак примене Хафове трансформације за кружнице је сличан претходно описаном поступку примене стандардне Хафове транформације. Потребно је извршити пресликавање сваке тачке из параметарског простора у Хафов простор помоћу формуле (55). Код Хафове трансформације за кружнице акумулатор се састоји од три параметра (a, b, r), где (a, b) представљају координате центра кружнице, а r полупречник те кружнице. Након пресликавања сваког детектованог ивичног пиксела у Хафов простор, врши се испитивање вредности акумулатора и издвајање оних ћелија које имају велику концентрацију пиксела. Након проналажења максималних вредности акумулатора, у параметарском простору се спајају тачке које припадају кружници полупречника r, а чији се центар налази у тачки (a, b).

8. Детекција дводимензионалних облика

Облик је један од основних извора информација који се користе за препознавање објекта. Широк је спектар подручја примене детекције облика, укључујући роботику, медицину, сигурносне системе и друго. На пример, у медицини се детекција облика може применити у циљу дијагностификовања болести у зависности од облика органа. Основни циљ детекције дводимензионалних облика јесте је идентификација облика објекта присутног у слици, при чему се облик дефинише као скуп контура које описују границе објекта.

Детекција основних дводимензионалних облика се састоји из три корака:

1. Детекција ивица у слици
2. Сегментација – раздвајање региона
3. Детекција облика

У литератури постоји доста техника за детекцију основних геометријских облика од којих је најједноставнија за примену и најчешће коришћена техника заснована на особинама региона.

У даљем делу овог рада је дат детаљнији опис наведене технике за детекцију облика у слици.

8.1. Техника заснована на особинама региона

Техника заснована на особинама региона једна од најпрецизнијих техника за детекцију облика региона у слици. Техника детекције облика заснована на особинама региона се састоји из три корака:

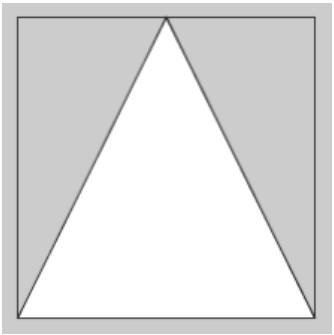
1. Издвајање ивица у слици – подразумева детекцију ивица објеката улазне слике применом неког од оператора за издвајање ивица у циљу једноставнијег раздвајања објеката
2. Сегментација слике – подразумева раздвајање објеката и попуњавање сваког региона

3. Детекција облика – подразумева идентификацију облика сваког од објекта односно региона у слици и детекцију оних објеката чији је облик неки од једноставних геометријских облика као што су круг, квадрат, правоугаоник или троугао.

Затворене контуре слике добијене претходно описаним издвајањем и повезивањем ивица у слици се називају региони. Сви региони имају неке особине, као што су површина, обим и центар, које се заправо односе на математичке карактеристике одређеног региона у слици [17]. На основу особина региона и његовог оквира врши се детекција облика региона.

Оквир региона (енг. *bounding box*) је замишљени правоугаоник који се користи за опис локације објекта. То је најмањи правоугаоник који у потпуности обухвата објекат чије су стране увек паралелне са осама.

На слици 90 је приказан пример креирања оквира региона [18].



Слика 90. Пример оквира региона

Поступак детекције облика се састоји од следећих корака:

- Детекција ивица у слици применом неког од описаних оператора
- Повезивање ивица у детектованој слици
- Попуњавање детектованих региона
- Одређивање карактеристика детектованих региона, као што су површина и обим
- Одређивање оквира региона (енг. *bounding box*)
- Израчунавање односа површине региона и оквира региона
- Класификација региона на основу израчунатог односа

У табели 1 је приказан начин класификације региона у зависности од односа површине и оквира региона [16].

Табела 1. Класификација облика у зависности од односа површине и оквира региона

Однос	Услов	Облик
0,25 < однос < 0,5	/	троугао
однос = 0,78	/	круг
однос != 1	/	непознат облик
однос = 1	ширина = дужина	квадрат
однос = 1	/	правоугаоник

У примеру детекције облика који ће бити приказан у даљем делу рада примењен је Собелов оператор за детекцију ивица у слици и Хафова трансформација за повезивање детектованих ивица. Након повезивања ивица коришћена је самостално осмишљена функција за попуњавање региона (написана у Матлабу) која је приказана на слици 91.

```
function [I] = fillRegion(I)
    for i=1:size(I,1)
        for j=1:size(I,2)-2
            if I(i, j) == 0 && I(i, j+1) == 1 && I(i, j+2) == 0
                for a = j+2:size(I,2)-1
                    if I(i, a) == 1 && I(i, a+1) == 0
                        I(i, j:a) = 1;
                        break
                    end
                end
            elseif I(i, j) == 0 && I(i, j+1) == 1 && I(i, j+2) == 1 ...
                    && I(i, j+3) == 0
                for a = j+4:size(I,2)-1
                    if I(i, a) == 1 && I(i, a+1) == 0
                        I(i, j:a) = 1;
                        break
                    end
                end
            elseif I(i, j) == 0 && I(i, j+1) == 1 && I(i, j+2) == 1 ...
                    && I(i, j+3) == 1 && I(i, j+4) == 0
                for a = j+5:size(I,2)-1
                    if I(i, a) == 1 && I(i, a+1) == 0
                        I(i, j:a) = 1;
                        break
                    end
                end
            end
        end
    end
end
```

Слика 91. Функција за попуњавање региона након повезивања ивица

Након попуњавања региона приступа се креирању оквира региона и рачунању површине региона и оквира региона. На слици 92 је приказан начин креирања оквира сваког од детектованих региона у слици.

```
I = image;
while sum(I(:)) > 0

    for i=1:size(I,1)-1
        for j=1:size(I,2)-1

            if I(i,j) == 1
                A = [i-1, j-1];
                B = [i-1, j+1];
                C = [i+1, j-1];
                D = [i+1, j+1];

                ab = false; ac = false; bd = false; cd = false;

                while ab == false || ac == false || cd == false || bd == false
                    while sum(I(A(1,1), A(1,2):B(1,2))) ~= 0
                        A=[A(1,1)-1, A(1,2)];
                        B=[B(1,1)-1, B(1,2)];
                    end

                    while sum(I(A(1,1):C(1,1), A(1,2))) ~= 0
                        A=[A(1,1), A(1,2)-1];
                        C=[C(1,1), C(1,2)-1];
                    end

                    while sum(I(C(1,1), C(1,2):D(1,2))) ~= 0
                        C=[C(1,1)+1, C(1,2)];
                        D=[D(1,1)+1, D(1,2)];
                    end

                    while sum(I(B(1,1):D(1,1), B(1,2))) ~= 0
                        B=[B(1,1), B(1,2)+1];
                        D=[D(1,1), D(1,2)+1];
                    end
```

Слика 92. Креирање оквира региона

Идеја за проналажење оквира региона јесте проналажење једног ивичног пиксела у слици и креирање оквира око пронађеног пиксела. У првом кораку оквир региона је величине 3×3. Након тога се врши провера да ли се ивице оквира региона секу са ивицама региона. Уколико се секу, оквир региона се шири за један пиксел у одговарајућем смеру све док не обухвати цео регион.

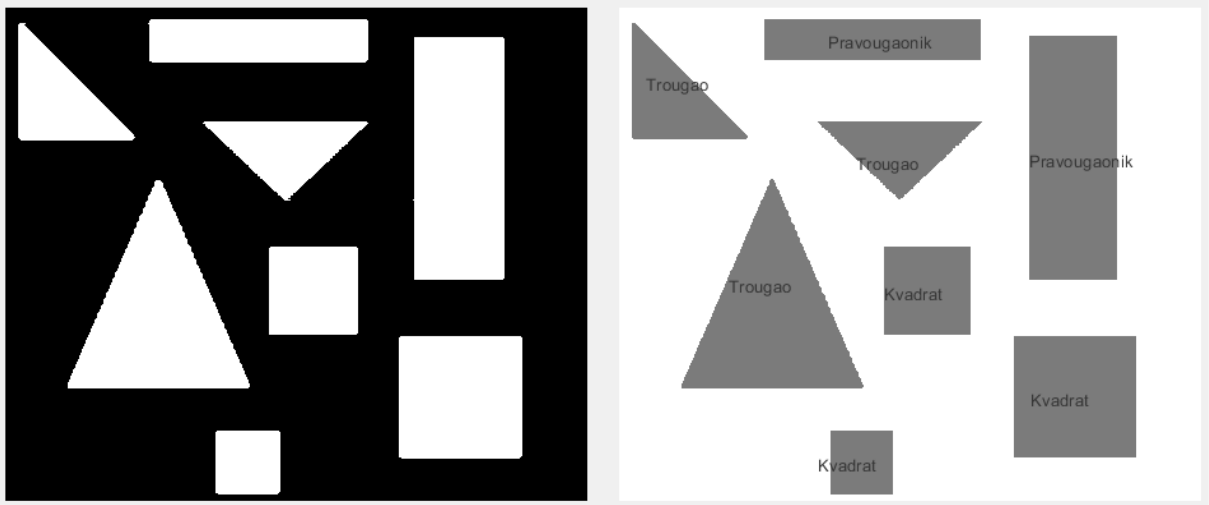
Након проналажења најмањег оквира региона на претходно описани начин, приступа се рачунању површине региона и оквира региона.

На слици 93 је дат приказ рачунања ових површина и упоређивања како би се добио коефицијент приближан вредностима из табеле 1 у циљу одређивања облика региона.

```
boxArea = (xmax-xmin)*(ymax-ymin);
shapeArea = 0;
for k=ymin:ymax
    for f=xmin:xmax
        shapeArea = shapeArea + I(k,f);
    end
end

I(ymin:ymax, xmin:xmax) = 0;
ratio = shapeArea/boxArea;

if 1 - ratio < 0.1
    if abs(xmax-xmin - (ymax-ymin)) <= 10
       shape(end+1) = 1; % square
    else
       shape(end+1) = 2; % rectangle
    end
elseif 1 - ratio < 0.26 && 1 - ratio > 0.18
    shape(end+1) = 4; % circle
elseif 1 - ratio < 0.8 && 1 - ratio > 0.46
    shape(end+1) = 3; % triangle
end
```

Слика 93. Одређивање облика региона у зависности од коефицијента односа површина

На слици 94 је приказан пример детекције облика на основу претходно описаног алгоритма и примене приказаних кодова.



Слика 94. Пример детекције облика региона у слици

9. Закључак

У овом раду је дат теоријски преглед различитих оператора за детекцију ивица у дигиталној слици, како градијентних, тако и Лапласових. Поред теоријског дела, приказана је и практична имплементација објашњених оператора у Матлабу. Из приказаних резултата се може закључити да најбоље резултате даје Канијев алгоритам у поређењу са осталим описаним операторима. Канијев алгоритам је најмање осетљив на шум и пружа добру локализацију ивица.

Поред приказа оператора за издвајање ивица, у раду је објашњен и на примеру приказан поступак спајања ивица коришћењем Хафове трансформације. Као додатак овом раду може се написати имплементација Хафове трансформације за кружнице, као додатак већ написаном коду за стандардну Хафову трансформацију.

Такође је објашњен и алгоритам за детекцију облика у слици који се заснива на особинама региона издвојених на основу претходно детектованих и повезаних ивица. Поменути алгоритам је имплементиран у Матлабу и прилагођен за детекцију основних облика као што су троугао, квадрат и правоугаоник.

Даље унапређење овог рада се огледа у испитивању других оператора за детекцију ивица у слици, додатних техника за детекцију облика, као и алгоритама за издвајање додатних обележја дигиталне слике, као што су текстура и боја.

10. Литература

[1] Електромагнетни спектар. Преузето 19.07.2020. са https://www.researchgate.net/figure/Electromagnetic-spectrum-80_fig16_318234623

[2] Поповић, В. М. (2006). *Дигитална обрада слике.* Београд: Академска мисао.

[3] Gonzales, C. R., Woods. E. R. *Digital image processing*. (2008). Pearson Education

[4] *Color spaces*. Преузето 23.07.2020. са https://aishack.in/tutorials/color-spaces-1/

[5] *Colour Image Processing.* Преузето 23.07.2020. са https://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/OWENS/LECT14/lecture12.html

[6] Astburys, A. (2013). *Резолуција у пикселима*. Преузето 29.07.2020. са https://www.wildlifeinpixels.net/blog/pixel-resolution/

[7] Shine, A. B. *Digital Image Fundamentals*. Преузето 02.08.2020. са https://www.slideshare.net/abshinde/digital-image-fundamentals-163572653

[8] Savant, S. *A Review on Edge Detection Techniques for Image Segmentation*. Преузето 03.08.2020. са https://www.researchgate.net/publication/291975278_A_Review_on_Edge_Detection_Techniques_for_Image_Segmentation

[9] Shubhashree, S. *A Review on Edge Detection Techniques for Image Segmentation*. Преузето 05.08.2020. са https://www.researchgate.net/publication/291975278_A_Review_on_Edge_Detection_Techniques_for_Image_Segmentation

[10] Owotogbe, J. *Edge Detection Techniques on Digital Images – A Review.* Преузето 05.08.2020. са https://www.researchgate.net/publication/338112798_Edge_Detection_Techniques_on_Digital_Images_-_A_Review

[11] Bradski, G., Kaehler, A. *Learning OpenCV*. (2008). O'Reilly Media

[12] Vicram, M. *Methods of Image Edge Detection: A Review*. Преузето 06.08.2020. са https://www.researchgate.net/publication/282775998_Methods_of_Image_Edge_Detection_A_Review

[13] Maini, R., Aggarwal, H. *Study and Comparison of Various Image Edge Detection Techniques*. Преузето 07.08.2020. са http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.301.927&rep=rep1&type=pdf

[14] Jackson, D. J. *Computer Vision & Digital Image Processing*. Преузето 29.08.2020. са http://jjackson.eng.ua.edu/courses/ece482/lectures/LECT17-2.pdf

[15] Ribeiro, B. M., Martins, D. A. & Neves, A. J. R. *Arbitrary ball detection using the circular Hough transform*. Преузето 08.09.2020. са https://www.researchgate.net/publication/242103081_Arbitrary_ball_detection_using_the_circular_Hough_transform

[16] Rege, S., Memane, R., Phatak, M., Agarwal, P. *2D geometric shape and color recognition using digital image processing*. Преузето 11.09.2020. са https://www.ijareeie.com/upload/june/48_2D%20GEOMETRIC.pdf

[17] Patel, S., Trivedi, P., Gandhi, V., Prajapati, G. I. *2D Basic Shape Detection Using Region Properties*. Преузето 18.09.2020. са https://www.ijert.org/research/2d-basic-shape-detection-using-region-properties-IJERTV2IS50606.pdf

[18] *Bounding-box of different shapes*. Преузето 20.09.2020. са https://www.geeksforgeeks.org/how-to-get-bounding-box-of-different-shapes-in-p5-js/