

SVEUČILIŠTE U ZAGREBU  
FAKULTET ORGANIZACIJE I INFORMATIKE  
VARAŽDIN

Filip Milohanović

IZRADA KOMPONENTE ZA DETEKCIJU  
OZNAČENIH ODGOVORA NA PISMENIM  
ISPITIMA

DIPLOMSKI RAD

Varaždin, 2025.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

**Filip Milohanović**

**JMBAG: 0016148270**

**Studij: Informacijsko i programsко inženjerstvo**

**IZRADA KOMPONENTE ZA DETEKCIJU OZNAČENIH ODGOVORA  
NA PISMENIM ISPITIMA**

**DIPLOMSKI RAD**

**Mentor :**

Doc. dr. sc. Marko Mijač

**Varaždin, lipanj 2025.**

*Filip Milohanović*

**Izjava o izvornosti**

Izjavljujem da je moj diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

*Autor potvrdio prihvaćanjem odredbi u sustavu FOI-radovi*

---

## **Sažetak**

Opsega od 100 do 300 riječi. Sažetak upućuje na temu rada, ukratko se iznosi čime se rad bavi, teorijsko-metodološka polazišta, glavne teze i smjer rada te zaključci.

**Ključne riječi:** riječ; riječ; ...riječ; Obuhvaća 7+/-2 ključna pojma koji su glavni predmet rasprave u radu.

# Sadržaj

<b>1. Uvod . . . . .</b>	<b>1</b>
<b>2. Metode i tehnike rada . . . . .</b>	<b>2</b>
<b>3. Razjašnjavanje problema . . . . .</b>	<b>3</b>
3.1. Kontekst i definicija problema . . . . .	3
3.2. Tehnička podloga i pristupi rješavanju problema . . . . .	4
3.3. Digitalna slika . . . . .	6
3.3.1. Slike u boji . . . . .	7
3.3.2. Sive slike . . . . .	8
3.3.3. Binarne slike . . . . .	9
3.3.4. Važnost različitih tipova digitalnih slika . . . . .	11
3.4. Digitalna obrada slike . . . . .	12
3.4.1. Preprocesiranje slike . . . . .	13
3.4.1.1. Pribavljanje slike . . . . .	13
3.4.1.2. Poboljšanje slike . . . . .	13
3.4.2. Procesiranje slike . . . . .	17
3.4.2.1. Segmentacija slike . . . . .	17
3.4.2.2. Morfološka obrada . . . . .	18
3.4.2.3. Reprezentacija i opis . . . . .	19
3.4.2.4. Detekcija i prepoznavanje objekata . . . . .	20
3.4.3. Postprocesiranje slike . . . . .	21
3.4.3.1. Obrada izvučenih podataka . . . . .	22
3.4.3.2. Vizualizacija rezultata . . . . .	22
3.4.3.3. Kompresija slike . . . . .	22
3.5. Analiza postojećih rješenja i pristupa problemu . . . . .	24
<b>4. Definiranje zahtjeva . . . . .</b>	<b>27</b>
4.1. Opis rješenja . . . . .	27
4.2. Funkcionalni zahtjevi . . . . .	28
4.3. Nefunkcionalni zahtjevi . . . . .	29
4.4. Slučajevi korištenja . . . . .	30
4.4.1. Slučaj korištenja: Ocjenjivanje ispita . . . . .	31
4.4.1.1. Korisnici . . . . .	31
4.4.1.2. Glavni scenarij . . . . .	31
4.4.1.3. Uključeni slučaj uporabe: Generiranje PDF matrice odgovora . . . . .	32

4.4.1.4. Uključeni slučaj uporabe: Unos slike matrice točnih odgovora . . . . .	32
4.4.1.5. Uključeni slučaj uporabe: Unos slike matrica odgovora za ocjenjivanje . . . . .	32
4.4.1.6. Uključeni slučaj uporabe: Detekcija odgovora . . . . .	32
4.4.1.7. Uključeni slučaj uporabe: Usporedba odgovora . . . . .	32
4.4.1.8. Ostali prošireni i uključeni slučajevi uporabe . . . . .	32
4.4.1.9. Tijek aktivnosti . . . . .	33
<b>5. Dizajn i izrada artefakta . . . . .</b>	<b>34</b>
5.1. Odabrane tehnologije . . . . .	34
5.2. Strukturni model sustava . . . . .	35
5.2.1. Klase ImageData i EmguCvImage . . . . .	37
5.2.2. Klase DetectedCircleBase i EmguCvCircle . . . . .	37
5.2.3. Klasa EmguCvImageProcessor . . . . .	37
5.2.4. Klase GradeCalculator i GradeScale . . . . .	38
5.2.5. Ostale klase . . . . .	38
5.3. Modeliranje ponašanja sustava . . . . .	39
5.3.1. Ponašanje sustava tijekom detekcije odgovora s kontrolnog ispita . . . . .	39
5.3.2. Ponašanje sustava tijekom ocjenjivanja . . . . .	41
5.3.3. Dijagram slijeda – Generiranje matrice odgovora . . . . .	42
5.4. GradeVision aplikacija . . . . .	43
5.4.1. Način rada . . . . .	43
5.5. GradeVision biblioteka . . . . .	43
<b>6. Evaluacija . . . . .</b>	<b>44</b>
6.1. Skup testnih podataka . . . . .	44
6.1.1. Prikaz koraka obrada slike nad testnim podatcima . . . . .	44
6.2. Manualno testiranje rješenja . . . . .	44
<b>7. Diskusija . . . . .</b>	<b>45</b>
7.1. Analiza rezultata . . . . .	45
7.2. Limitacije rješenja . . . . .	45
7.3. Poboljšanja rješenja . . . . .	45
<b>8. Zaključak . . . . .</b>	<b>46</b>
<b>Popis literature . . . . .</b>	<b>49</b>
<b>Popis slika . . . . .</b>	<b>50</b>
<b>Popis popis tablica . . . . .</b>	<b>51</b>

# 1. Uvod

U današnjem obrazovnom sustavu nastavnici provode jako puno vremena na radnim zadacima koji ne podižu kvalitetu obrazovanja već su administrativni i često repetitivni poslovi. Jedan od tih zadataka je ocjenjivanje ispita, pogotovo ako se radi o velikoj količini pristupnika testu. Kod testova s izrazito velikom količinom pristupnika se zato koriste ispitni koji uključuju list za odgovore što olakšava ocjenjivanje. Time se značajno ubrzava cijeli proces ocjenjivanja pošto nije potrebno čitati odgovore koji su u različitim formatima i potencijalno na više stranica, ali i dalje ostaje repetitivni dio posla koji uključuje usporedbu odgovora s rješenjima. Kroz ovaj rad bit će prikazana softverska komponenta otvorenog koda koja rješava ovaj problem tako da naspram slike detektira označene odgovore i ocjenjuje ispit.

Kroz ovaj rad istražuje se razne metode za detektiranje označenih odgovora s ciljem izrade komponente visoke preciznosti, prateći metodologiju znanstvenog oblikovanja.

## 2. Metode i tehnike rada

S obzirom na to da se radi o temi koja uključuje obradu slike, što zahtijeva opsežno istraživanje, manipulaciju parametrima i testiranje pristupa odlučeno je koristiti inkrementalan pristup razvoju. Prije same izrade rješenja, provedena je minimalna potrebna količina istraživanja, a zatim se ostatak istraživanja odvijao paralelno s razvojem.

Iterativni razvoj ovog rješenja može se podijeliti u 3 faze: preprocesiranje, procesiranje i postprocesiranje slike. Jedna od najvažnijih tehnika tijekom razvoja bila je vizualizacija svakog koraka obrade slike. To je omogućilo izravan uvid u status i smjer razvoja. Time je od samog početka omogućen uvid u mane i prednosti korištenih metoda. Ovo je značajno doprinijelo definiranju budućih razvojnih koraka, poput dodatnog istraživanja, promjene pristupa i slično.

Da bi sama vizualizacija bila što učinkovitija korišteno je više različitih fotografija, slikanih u različitim uvjetima. To je omogućilo verifikaciju funkcionalnosti dijelova rješenja na različitim ulaznim podacima od samog početka razvoja, što je značilo da na kraju projekta nije bilo potrebno pokrivati rubne slučajeve jer je rješenje bilo dizajnirano da ih pokriva od samog početka.

Važno je napomenuti da je ovaj rad izrađen koristeći pristup znanstvenog oblikovanja ((eng. *design science*)). Znanstveno oblikovanje temelji se na razvoju i evaluaciji praktičnih rješenja, to jest artefakata. Razvijeni artefakt u ovom radu je softverska biblioteka otvorenog koda za detekciju odgovora na pismenim ispitima. Cilj tog artefakta je omogućiti fleksibilno ocjenjivanje pismenih ispita u različitim uvjetima. Sam razvoj pratio je iterativni proces uz stalnu evaluaciju pomoću vizualizacije rezultata i testiranja na raznolikim ulaznim podacima. To je ujedno bila metodologija znanstvenog oblikovanja ovog rada, koja je bila prisutna u svim fazama razvoja: dizajnu, implementaciji, testiranju i prilagodbi rješenja.

Rješenje je izrađeno pomoću .NET tehnologije i C# programskog jezika u obliku biblioteke, što omogućuje drugim rješenjima jednostavnu integraciju koristeći .dll.

Kroz ovaj rad nije bio cilj razviti vlastitu biblioteku za računalni vid, već je korištena jedna od postojećih biblioteka otvorenog koda. Osim toga, rješenje omogućuje korisnicima da koriste alternativnu biblioteku s uvjetom da izrade svoju implementaciju odgovarajućih metoda. Detaljniji razlozi odabira biblioteke prikazani su u nastavku rada.

### **3. Razjašnjavanje problema**

Kroz teoretski dio rada predočit će se problem, tehnologije i alati za njegovo potencijalno rješavanje, kao i cijeli proces detekcije odgovora na pismenim ispitima korištenjem odabralih tehniku. Također će se analizirati postojeća rješenja ovog problema.

Važno je napomenuti da postoje različiti pristupi rješavanju ovog problema, no svi ti pristupi imaju isti cilj. Taj cilj je izvlačenje relevantnih podataka iz slike te naknadna obrada tih podataka. U kontekstu ovog rada to znači da se iz slike moraju izvući podaci o označenim odgovorima za određeno pitanje te se ti podaci uspoređuju s listom točnih odgovora.

#### **3.1. Kontekst i definicija problema**

Ocenjivanje ispita na papiru jedan je od procesa koji učiteljima, nastavnicima i profesorima oduzima značajan dio vremena, koje bi inače mogli posvetiti drugim aktivnostima, poput izrade kvalitetnijih nastavnih materijala i sličnog. Taj problem postaje još izraženiji kada se radi o velikom broju ispita koje treba ocijeniti. Upravo zbog tih izazova, ispiti namijenjeni većem broju pristupnika često se standardiziraju tako da se pitanja prilagode [1].

Najčešće ta prilagodba podrazumijeva uklanjanje pitanja koja zahtijevaju tekstualni upis, povezivanje pojmove, crtanje i slično. Takva se pitanja uglavnom izostavljaju iz ispita za masovno ocjenjivanje jer predstavljaju podatke koje je teže obraditi ili zahtijevaju razumijevanje. Zbog toga se koriste isključivo tipovi pitanja koji imaju unaprijed definiranu strukturu i jednoznačno definiran odgovor ili odgovore. Time se problem ocjenjivanja ispita svodi na detekciju označenih odgovora i njihovu usporedbu s točnim odgovorima. Taj tip ispita značajno ubrzava sam proces ocjenjivanja kada ga provode ljudi, no može se još značajnije ubrzati koristeći sustave za automatizirano ocjenjivanje [2].

Sustavi za automatizirano ocjenjivanje koriste različite tehnologije kako bi omogućili učinkovito ocjenjivanje ispita polaznika. Neke od najčešće korištenih tehnologija uključuju obradu slike, strojno učenje te, u posljednje vrijeme, umjetnu inteligenciju. Među glavnim prednostima ovih sustava ističu se brzina ocjenjivanja, konzistentnost i objektivnost, što je teško postići kada više ocjenjivača ocjenjuje stotine ili tisuće ispita [3].

Ovi sustavi mogu ocjenjivati različite vrste ispita, ovisno o tipu pitanja. Najčešće se primjenjuju za ocjenjivanje pitanja tipa točno/netočno, jednostrukog i višestrukog odabira. U posljednje vrijeme sve se češće koriste i za ocjenjivanje esejskih pitanja, što je omogućeno brzim razvojem umjetne inteligencije [3].

Unatoč navedenim prednostima, postoje i određene prepreke. Jedna od najvećih je ocjenjivanje odgovora u kojima je važna inovativnost, kreativnost i kvaliteta odgovora. To se uglavnom odnosi na esejska pitanja i problemske zadatke. Danas se čak i takvi odgovori mogu ocjenjivati pomoću umjetne inteligencije, no to područje je i dalje u fazi razvoja i suočeno je s brojnim otvorenim pitanjima. Jedan od ključnih izazova je sposobnost razumijevanja složenih zadataka i konteksta, kao i problemi poput halucinacija modela [3].

Osim tehničkih izazova, postoji i potreba za edukacijom ocjenjivača za pravilno korištenje ovakvih sustava, kao i za provedbu mjera za prevenciju plagijata [3].

Ipak, kada se radi o ocjenjivanju ispita u kojima je kriterij isključivo točnost odgovora, bez potrebe za analizom kreativnosti, može se reći da su takvi sustavi već dovoljno zreli za pouzdanu primjenu. Takvi sustavi moraju detektirati odgovore s ispita i usporediti ih s točnim odgovorima.

Prvi takav uređaj bio je Type 805, kojeg je razvio IBM 1937. godine. Taj uređaj funkcionirao je tako da je provjeravao provodljivost napona na papiru. Točnije ako je vodljivost bila veća, smatralo se da je taj dio papira označen olovkom, budući da grafit provodi struju [4].

Zatim je 1950-ih razvijena i optička varijanta sustava koja je provjeravala količinu reflektiranog svjetla. Ta tehnologija kasnije je nazvana OMR (eng. Optical Mark Recognition). S razvojem računala postupno se smanjila potreba za posebnim uređajima s OMR tehnologijom, jer se ista tehnologija mogla koristiti pomoću standardnog računala i uređaja za dohvata sliku, poput skenera [4].

Iako ljudima proces detekcije odgovora i njihove usporedbe s točnim odgovorima uglavnom ne predstavlja poteškoće, kod računala je situacija znatno složenija. Ljudski mozak sposoban je prepoznati složene uzorke gotovo podsvjesno i u vrlo različitim uvjetima, dok je računalu takav zadatak izrazito zahtjevan. Zbog toga sama detekcija odgovora predstavlja jedan od najvećih izazova u razvoju sustava za automatizirano ocjenjivanje.

Jedan od glavnih problema su razlike u načinu na koji je ispit fotografiran ili skeniran. Tu se ubrajaju faktori poput udaljenosti uređaja od papira, kuta snimanja, neujednačenog osvjetljenja, sjena, mrlja, kao i kvalitete senzora koji je pribavio sliku. Sve te razlike mogu značajno utjecati na točnost prepoznavanja označenih odgovora [4].

Osim tehničkih izazova, dodatnu složenost unosi i činjenica da korisnici često ne slijede upute o označavanju odgovora, što može dodatno otežati interpretaciju. Zbog kombinacije tih problema važno je razvijati sustave koji su dovoljno robustni da prepoznaju odgovore i u ne-povoljnim uvjetima, ali i dovoljno fleksibilni da podnose razlike u načinu ispunjavanja obrazaca [4].

### 3.2. Tehnička podloga i pristupi rješavanju problema

Do sada su spomeuti početci OMR tehnologije, no za ovaj rad su relevantne nove tehnike i tehnologije vezane za detekciju odgovora na pismenim ispitima. Činjenica je da je OMR tehnologija započela kao obrada slike no danas je nadograđena raznim drugim tehnikama poput računalnog vida, strojnog učenja i umjetnom inteligencijom. Svaka od navedenih tehnika pomaže riješiti određene izazove s kojima se tradicionalno OMR suočavao.

Važno je napomenuti da se ove tehnike u različitoj literaturi često tretiraju kao jedinstvena tehnika, odnosno da se nerijetko koristi jedan podpojam, iako se zapravo misli na širi skup povezanih metoda. Primjerice, obrada slike tehnički spada pod računalni vid, no u mnogim radovima koristi se izraz "obrada slike" i kada se misli na šire postupke karakteristične za

računalni vid. Slična je situacija i sa strojnim učenjem i umjetnom inteligencijom, gdje se strojno učenje često koristi kao sinonim za AI, iako je ono zapravo njezin podskup. Kroz ovaj rad će se također koristiti termin "obrada slike" u širem smislu, pri čemu će se pod time podrazumijevati i tehnike računalnog vida.

Tradicionalno je OMR tehnologija zahtijevala specijalizirano i skupo sklopolje, no danas se uz pomoć računalnog vida i obrade digitalne slike cijeli proces može pojednostaviti. Slika se ne pribavlja skupim uređajima već mobitelom ili skenerom, što čini cijeli proces pribavljanja slike puno jeftinijim, no ne uklanja probleme vezane za kvalitetu slike koje smo spominjali u prijašnjem poglavlju. Da bi se ti problemi riješili, koristi se računalni vid i obrada slike koji nude mogućnosti smanjivanja šuma, ispravka perspektive i slično. Ovim tehnikama se može softverski implementirati efikasniji i jeftiniji OMR sustav. No takvi sustavi imaju ograničenje da ne mogu obrađivati tekstualna pitanja, već samo detektiraju oznake [5].

Da bi se podržala i takva pitanja, može se koristiti umjetna inteligencija, to jest strojno učenje, većinom sustavi za obradu prirodnog jezika NLP (eng. natural language processing). No, kao što je spomenuto u prijašnjem poglavlju, umjetna inteligencija je još uvijek jako svježa tehnologija i razina zrelosti za konzistentno i pouzdano ocjenjivanje je i dalje upitna zbog raznih problema poput halucinacija. Prema nekim istraživanjima takvi sustavi imaju otprilike 85% točnosti. Točnost će kroz vrijeme sigurno rasti, no treba odgovoriti i na potencijalna etička pitanja [6].

Kroz ovaj rad nije cilj razviti vlastitu biblioteku za računalni vid, već će se koristiti već postojeće biblioteke ili okvire. Kroz priloženu tablicu prikazani su najpopularnije biblioteke i okviri, važno je napomenuti da se radi o bibliotekama i okvirima otvorenog koda.

Biblioteka	Ključne značajke	Prednosti	Nedostaci
<b>OpenCV</b>	2,500+ algoritama, podrška za više platformi	Optimiziran, GPU ubrzanje	Nema ugrađenu podršku za duboko učenje
<b>Scikit-Image</b>	Obrada slike s NumPy integracijom	Jednostavan, idealan za početnike	Nije optimiziran za duboko učenje
<b>TensorFlow</b>	Modeli dubokog učenja, TensorFlow Lite za mobilne uređaje	Skalabilan, produksijski spreman	Strma krivulja učenja
<b>PyTorch</b>	TorchScript za izvođenje modela izvan Python-a, unaprijed obučeni modeli	Fleksibilan, jednostavno za korištenje i debugiranje	Manje skalabilan od TensorFlow-a
<b>OpenVINO</b>	Duboko učenje, Optimizirano za Intel sklopovlje	Visoka učinkovitost na Intel čipovima	Ograničen na Intel platforme
<b>Detectron2</b>	Napredna segmentacija i detekcija objekata	Vrlo precizan, podrška za PyTorch	Fokusiran samo na detekciju

Tablica 1: Usporedba biblioteka i okvira računalnog vida [7]

Iz priložene tablice mogu se vidjeti značajke, prednosti i nedostaci različitih biblioteka i

okvira za računalni vid. Neke, poput OpenCV-a i Scikit-Image, podržavaju samo tradicionalni računalni vid i obradu slike bez korištenja dubokog učenja, dok se druge, poput TensorFlow-a i PyTorch-a, fokusiraju samo na duboko učenje. Osim razlika u domeni primjene, postoje i razlike u programskim jezicima koje biblioteke podržavaju, kao i u sklopolju potrebnom za njihov rad. Inače je riječ je o vrlo popularnim alatima koji značajno olakšavaju razvoj softvera temeljenog na računalnom vidu [7].

Pošto će se ovaj rad više fokusirati na računalni vid temeljen na klasičnoj obradi slike, a ne na dubokom učenju, iz priložene tablice u obzir dolaze samo OpenCV i Scikit-Image biblioteke. U praktičnom dijelu rada bit će objašnjeno koja je biblioteka odabrana i koji su razlozi za njezin konačni odabir.

Prije prikaza konkretnih tehnika i tehnologija koje se u tu svrhu koriste, potrebno je definirati nekoliko osnovnih pojmoveva poput digitalne slike.

### 3.3. Digitalna slika

Za početak važno je razumjeti što su to digitalne slike i koje sve informacije one sadrže. Digitalna slika je zapravo računalna datoteka koja reprezentira fotografiju pomoću piksela. Piksela je zapravo najmanji element slike koji je reprezentiran s 3 kanala boja: Crvena (R), zelena (G), plava (B) u RGB modelu. Postoje i drugi kanali boja, ali za potrebe ovog rada fokus će biti na RGB modelu. [8].

Sama boja piksela se određuje kombinacijom vrijednosti iz RGB kanala. Dok je sam broj mogućih boja određen brojem bitova koji svaki RGB kanal može poprimiti. Taj koncept se još naziva dubinom boje (*eng. color depth*). Na primjer, pikseli 24-bitne slike mogu poprimiti do 16,777,216 unikatnih boja pošto svaki kanal ima 8 bitova. [8]:

$$\text{Broj boja} = 2^8 \times 2^8 \times 2^8 = 2^{24} = 16.777.216$$

S pomoću piksela se također može definirati veličinu slike, to jest rezolucija. Rezolucija je sveukupan broj piksela koje neka slika sadrži i izražava se množenjem širine slike s visinom slike:

$$\text{Ukupan broj pikslea} = \text{Širina} \times \text{Visina}$$

Za sliku rezolucije  $1920 \times 1080$  to je:

$$\text{Ukupan broj pikslea} = 1920 \times 1080 = 2.073.600 \text{ piksela}$$

Obično veća rezolucija znači više detalja na slici. Što se više detalja nalazi na slici to su bolje šanse za izvući korisne informacije sa slike. No naravno da rezolucija sama ne određuje kvalitetu digitalne slike, već veliku ulogu igraju i fizički uvjeti u kojima je slika izrađena. Pod to spada osvjetljenje, čistoća kamere, kvaliteta senzora i sl.

Sliku se osim metrika poput dubine boja i rezolucije može opisati i omjerom slike (*eng. aspect ratio*). Omjer slike opisuje odnos broja piksela po širini i visini slike. Omjer slike većinom je vezan za način na koji je slika uslikana ili na način na koji se ona reproducira. Na primjer, profesionalni fotoaparati većinom koriste omjer slike 3:2, kamere pametnih telefona koriste omjer 4:3 dok računalni monitori koriste 16:9 [9].

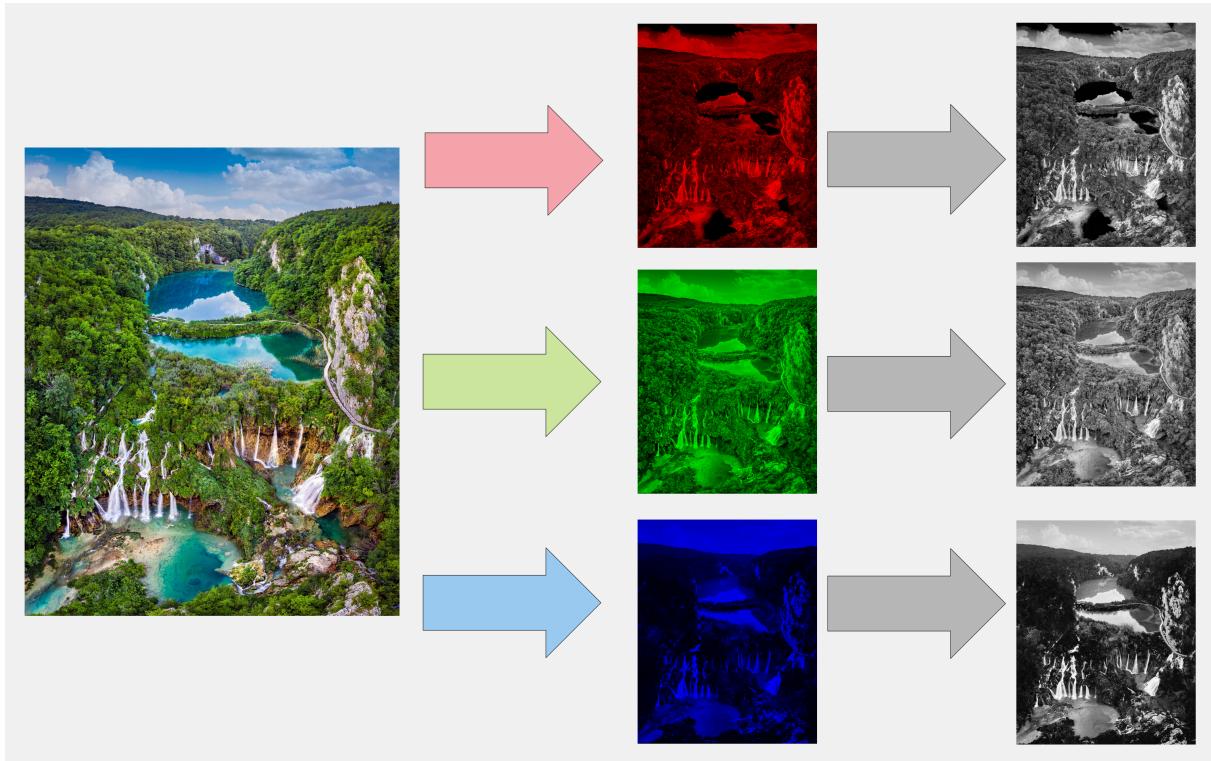
Važno je napomenuti da se većina do sada spomenutih pojmoveva odnosi isključivo na slike rasterske grafike, dok za vektorske slike vrijede neki drugi pojmovi i pravila. U ovom radu koristit će se rasterske slike, odnosno slike koje se sastoje od piksela. Za razliku od rasterske grafike, vektorska grafika nije bazirana na pikselima, već je definirana matematičkim funkcijama i krivuljama. To ujedno znači da za nju ne vrijedi pojmom rezolucije, budući da njezina kvaliteta nije ograničena statičnim informacijama poput piksela, već se dodatni detalji mogu izračunati iz definiranih funkcija [8].

### 3.3.1. Slike u boji

Povijest fotografije počinje 1830-ih godina, prve fotografije su bile crno-bijele. No to se već 1861. promjenilo pojavom prve slike u boji. Clerk Maxwell je korištenjem crvenih, zelenih i plavih filtera izradio prvu sliku u boji. Zatim je kompanija Kodak popularizirala slike u boji i danas su one de facto standard. Zatim se s vremenom analogna fotografija razvila u digitalnu fotografiju [10].

Danas većina digitalnih slika koje ljudi stvaraju sadrže boje i koriste sva tri kanala boje. Za razumijevanje ovog rada potrebno je razumjeti kakve informacije sadrži pojedini kanal boje. Svaki kanal predstavlja jednu primarnu boju poput crvene, zelene ili plave. I svaki piksel prima određenu vrijednost za svaki kanal, ovisno o kombinaciji bitova. Za 24-bitnu sliku svaki kanal ima 8 bitova to jest 256 mogućih vrijednosti.

To znači da u svakom kanalu postoji vrijednost od 0 do 255 koja definira jačinu svjetlosti za tu primarnu boju. Nulu bi označavala kompletno crna boja dok bi 255 bila potpuno osvjetljena primarna boja, ovisno o kojem se kanalu radi. Kombinacijom tih 3 kanala se zapravo dobivaju međuboje i time se stvara konačni izgled slike. [11].



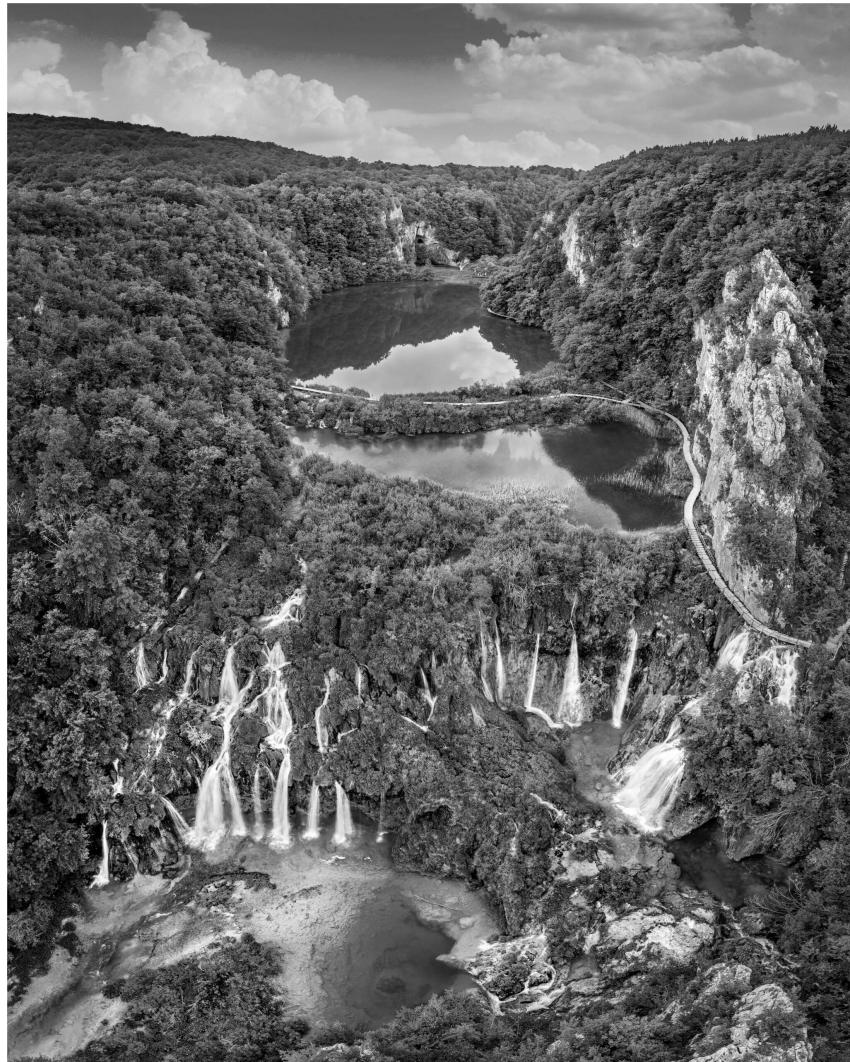
Slika 1: Prikaz slike u boji i pripadajućih kanala boja u RGB modelu (vlastita izrada)

Kao što je prikazano na slici, kanali boja prikazani su kao slike čiji pikseli poprimaju crnu, bijelu i nijanse sive boje. Također kanali se ponekad znaju vizualizirati pomoću primarne boje kanala kojeg reprezentiraju, ali sami podaci koje digitalna slika sadrži su uvijek jednaki, jedino način njihove vizualizacije varira.

### 3.3.2. Sive slike

Sive slike (*eng. grayscale*) su poseban tip digitalnih slika koji ima samo jedan kanal boja. U tom kanalu vrijednost piksela reprezentira svjetlinu piksela. Vizualizacija takvih slika poprima boje od bijele do crne uključujući nijanse sive. Na primjer za sliku s dubinom boja od 8 bitova, nula predstavlja crnu boju, 255 prestavlja bijelu boju, a međuvrijednosti su predstavljene nijansama sive boje [11].

Sive slike su jedan od važnijih koncepata obrade slike i računalnog vida. One su zapravo početna točka kod mnogih algoritama za procesiranje slika. Često se koriste kao prvi korak obrade. Izrazito su korisne za detekciju rubova, segmentaciju slike, prepoznavanje uzoraka i slično [12]. Kroz rad će se detaljnije prikazati njihova važnost i uloga u kontekstu detektiranja označenih odgovora na pismenim ispitima.



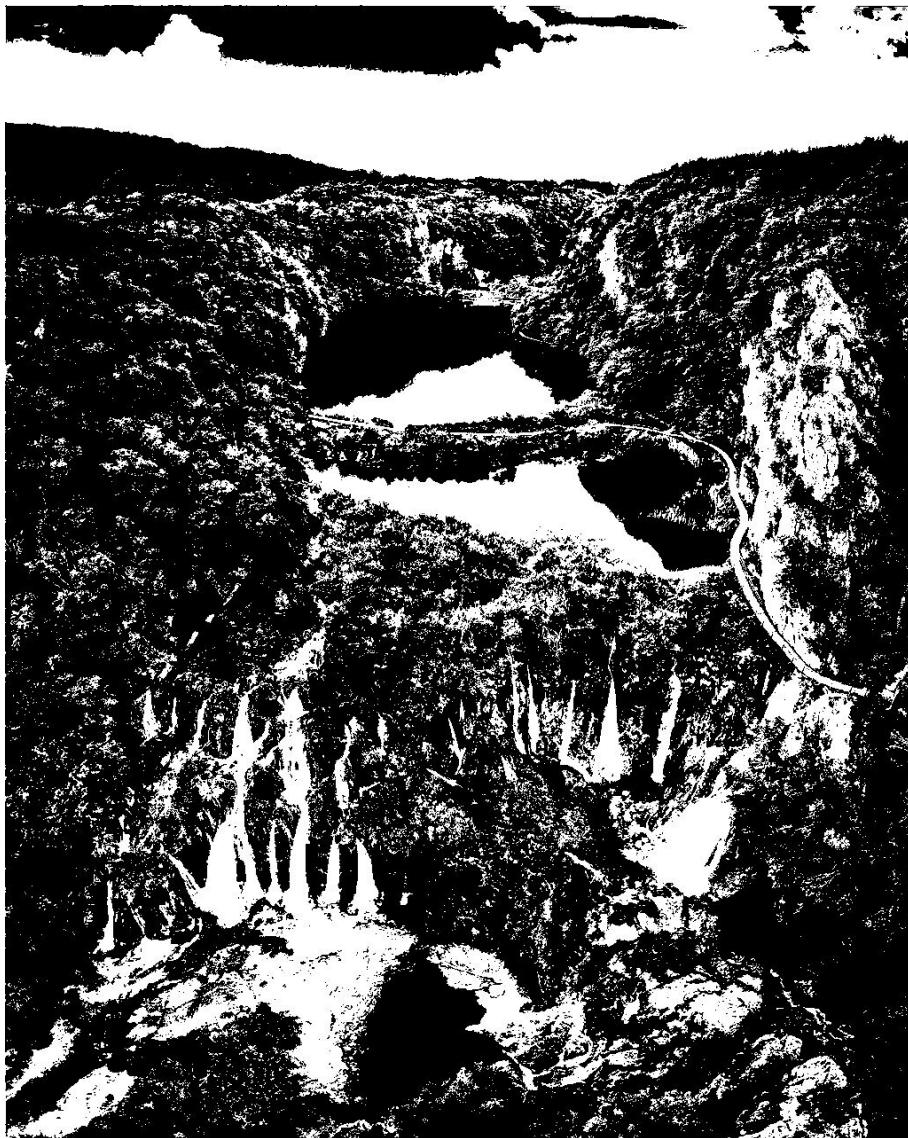
Slika 2: Prikaz sive slike (vlastita izrada)

Na priloženoj sivoj slici se također može vidjeti da je različita od sivih slika kanala prikazanih na slici 1. To je zato što se kod pretvorbe slike u boji u sivu sliku uzimaju u obzir vrijednosti piksela za sva 3 kanala prema formuli:

$$I_{\text{siva}}(x, y) = 0.299 \cdot R(x, y) + 0.587 \cdot G(x, y) + 0.114 \cdot B(x, y)[12]$$

### 3.3.3. Binarne slike

Osim slika u boji i sivih slika također postoje i binarne slike. Binarne slike su poseban tip slika gdje piksel može imati samo dvije moguće vrijednosti od kud im dolazi i naziv, te vrijednosti su nula ili jedan. Binarne slike sadrže puno manje informacija od ostalih tipova slika, ali isto tako smanjuju kompleksnost same slike i omogućuju fokusiranje na važne značajke slike. Baš zbog toga se jako često koriste kod obrada slika koristeći računalni vid [13].



Slika 3: Prikaz binarne slike (vlastita izrada)

Sam proces izrade binarne slike je dosta jednostavan. Za početak potrebno je imati sivu sliku. Zatim se definira granica (*eng. threshold*) u obliku postotka ili vrijednosti piksela. Ta granica se zatim koristi za određivanje nove vrijednosti piksela, ako je vrijednost iznad granice onda se vrijednost postavlja na jedan, a inače na nulu. Priložena formula prikazuje osnovni tip kreiranja binarne slike (*eng. thresholding*).

$$I_{\text{siva}}(x, y) = 0.299 \cdot R(x, y) + 0.587 \cdot G(x, y) + 0.114 \cdot B(x, y)$$

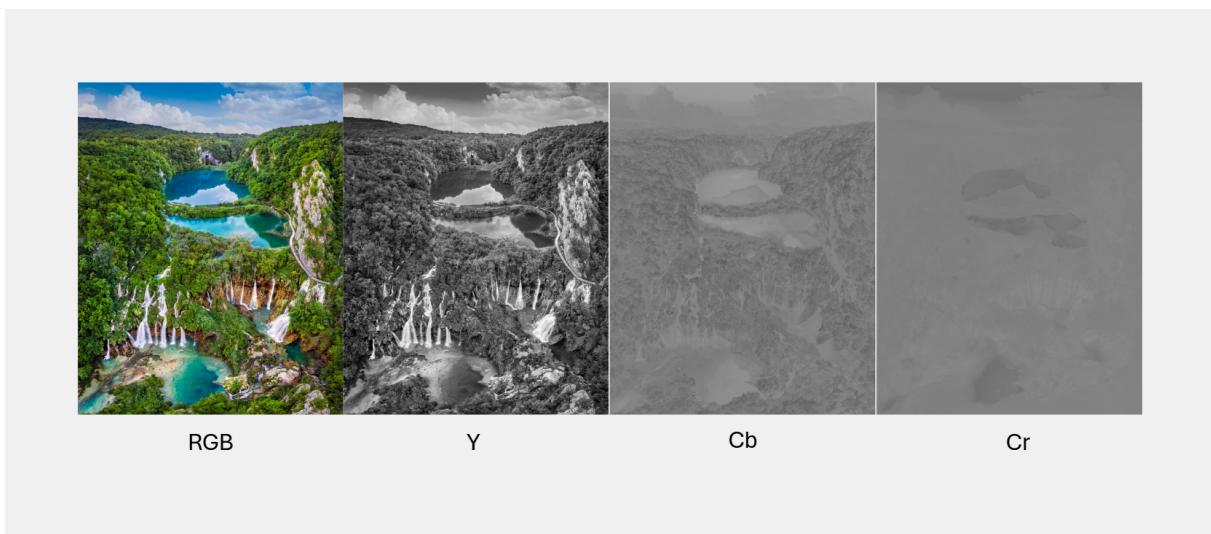
$$I_{\text{binarna}}(x, y) = \begin{cases} 1, & \text{ako } I_{\text{siva}}(x, y) \geq T \\ 0, & \text{ako } I_{\text{siva}}(x, y) < T \end{cases}$$

### 3.3.4. Važnost različitih tipova digitalnih slika

Do sada smo prošli kroz 3 najvažnija tipa digitalnih slika za ovaj rad. Kroz ovu cjelinu prikazat će se važnost pojedinog tipa slike naspram ostalih.

Zapravo najveća razlika između ovih tipova je broj kanala boje. Slike u boji imaju 3 kanala za razliku od sivih i binarnih slika koje imaju po jedan kanal. To zapravo znači da slike u boji sadrže značajno više informacija.

No ponekad nam te informacije nisu potrebne ovisno o problemu koji pokušavamo riješiti. Na primjer ako pokušavamo detektirati boju svjetla na semaforu onda su nam informacije o bojama izrazito važne. No ako pokušavamo pročitati tekst s prometnog znaka onda nam informacija o boji znaka i teksta ne doprinosi značajno rješavanju problema. Zapravo nam otežava rješavanje problema pošto moramo raditi s nepotrebnim informacijama.



Slika 4: Usporedba svjetlosne komponente i komponenta boja(vlastita izrada)

Na priloženoj slici prikazana je ista slika na različite načine:

- Slika u RGB formatu
- Slika u YCbCr formatu
  - Svjetlosne informacije u Y kanalu
  - Informacije o bojama u Cb i Cr kanalima

Na ovom primjeru je prikazana činjenica da svjetlosni kanali sadrže puno više informacija o strukturalnim elementima na slici. Baš zbog toga se u raznim algoritmima za detektiranje značajki, filtriranje, segmentiranje i sl. koriste baš sive slike [14].

Sive slike se koriste kada boja nije važan faktor u obradi slike. U tom slučaju je boja jednako korisna kao buka na slici pa je zbog toga uklonimo. Time se veličina slike smanjuje i ujedno se ubrzava daljnji proces obrade slike [14].

Ako je potreban još agresivniji pristup otklanjanju nepotrebnih podataka, binarne slike mogu biti korisne. One dodatno odstranjuju nepotrebne podatke tako da povećavaju razliku u kontrastu i time odvajaju razne strukturne elemente slike [14].

Važno je napomenuti da svaki tip slika ima svoju upotrebu ovisno o problemu koji se rješava. U ovom radu će veliku važnost imati sive i binarne slike pošto se radi o procesu detektiranja strukture fotografije i izvlačenja relevantnih informacija iz nje.

### 3.4. Digitalna obrada slike

Digitalna obrada slike (*eng. digital image processing*) podrazumijeva obradu digitalne fotografije koristeći računalne algoritme s ciljem unaprjeđivanja slike, izvlačenja korisnih informacija, analize, izrade izvještaja i slično. Predstavlja ključni korak u raznim primjenama računalnogvida temeljenog na dubokom učenju, poput prepoznavanja lica, detekcije objekata i optičke detekcije [15].

Sam proces digitalne obrade slike može se podijeliti u sljedeće faze [15]:

- Pribavljanje slike (*eng. image acquisition*)
- Poboljšanje slike (*eng. image enhancement*)
- Kompresija slike (*eng. image compression*)
- Morfološka obrada (*eng. morphological processing*)
- Segmentacija slike (*eng. image segmentation*)
- Reprezentacija i opis (*eng. representation and description*)
- Detekcija i prepoznavanje objekata (*eng. object detection and recognition*)

Naravno nisu sve faze uvijek potrebne, već se radi o generalnim smjernicama koje se prema specifičnostima problema mogu modificirati, smanjiti ili proširiti. U nekim slučajevima čak se redoslijed faza može promijeniti ili se faze mogu provoditi više puta.

Te faze se dodatno mogu podijeliti na **preprocesiranje, procesiranje i postprocesiranje slike**. U fazu **preprocesiranja** spadaju svi koraci koji pripremaju sliku za procesiranje, poput poboljšanja slike. Nakon preprocesiranja slijedi **procesiranje** koje se odnosi na korištenje algoritama za detekciju i prepoznavanje objekata i transformacije. Posljednja faza uključuje korake poput pripreme rezultata za prezentaciju, vizualizaciju i daljnju analizu [16].

### **3.4.1. Preprocesiranje slike**

Preprocesiranje slike je ključan proces koji se provodi prije obrade slike. Kao ulaz prima izvornu, sirovu sliku, a kao izlaz vraća sliku u korisnjem formatu za daljnju obradu i analizu. Omogućuje da se sa slike odstrane ili istaknu područja relevantnih informacija te da se poboljša kvaliteta slike prije daljnje obrade [17].

#### **3.4.1.1. Pribavljanje slike**

Pribavljanje slike je proces nastajanja digitalne slike tako da se informacije iz realnog svijeta zapišu u digitalnom obliku kojim računalo zatim može manipulirati. Slika se može pribaviti raznim uređajima poput fotoaparata, pametnog mobitela i drugih tehnologija poput skenera, rendgena. U ovom radu fokus će biti na fotografije pribavljene uređajima koji posjeduju digitalnu kameru [18].

Da bi bilo kakvo procesiranje moglo krenuti, prvo je potrebno pribaviti sliku. Moglo bi se reći da je pribavljanje slike čak najvažniji korak u cijelom procesu obrade slike, njegov rezultat ima veliku ulogu za postizanje krajnjeg rezultata obrade i analize slike. Pribavljena slika mora biti kvalitetna da bi ostali koraci bili uspješni. Na kvalitetu slike ne utječe samo rezolucija, već osvjetljenje i kut pod kojim je slika uslikana također imaju značajnu ulogu [18].

Razni algoritmi i koraci u preprocesiranju mogu korigirati pribavljenu sliku, no to nikad nije zamjena za sliku dobre kvalitete. Dobra i loša slika mogu biti razlika između uspješnog i neuspješnog detektiranja elemenata na slici. Naravno u većini slučajeva sustavi za obradu slike ne mogu utjecati na kvalitetu slike koje će im korisnik poslati. Zbog toga je važno detaljno testirati sustav u različitim uvjetima i utvrditi što su njegove granice i je li potrebno podržati neki dodatni korisnički slučaj od onih postojećih.

#### **3.4.1.2. Poboljšanje slike**

Poboljšanje slike je ključan korak, cilj mu je modificirati sliku tako da ona postane korisnija za daljnju obradu. To se može postići na više načina poput otklanjanja buke, isticanja važnih značajki slike, poboljšanja oštchine i kontrasta slike.

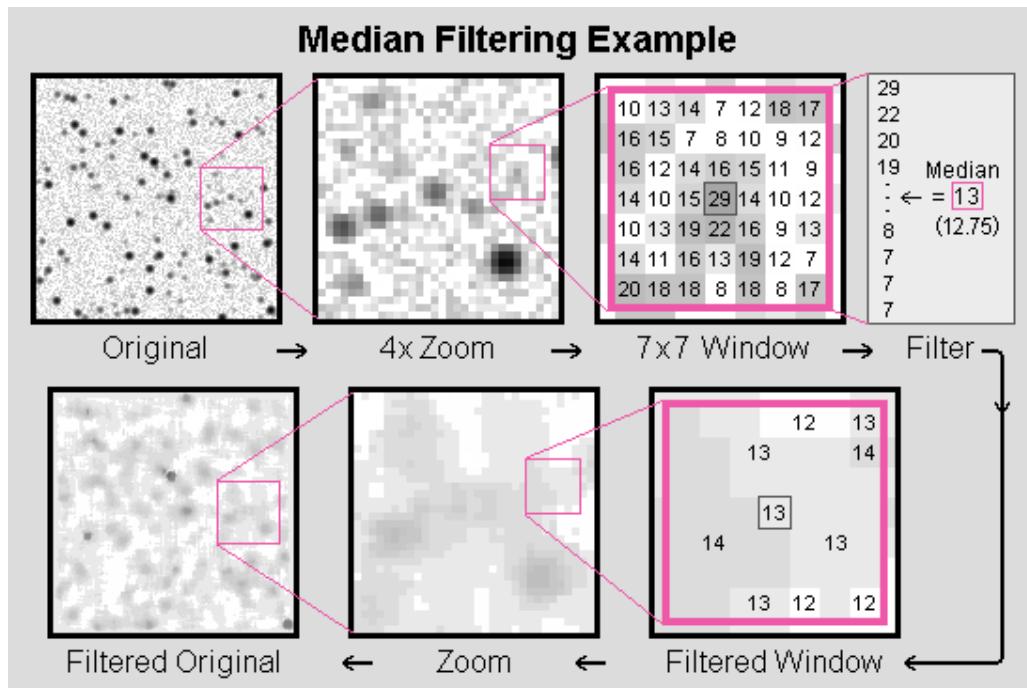
Do sada smo spomenuli da se nepotrebne informacije na slici također mogu biti buka. Baš zbog toga pretvaranje u sivu i binarnu sliku spada pod poboljšanje slike. Oba dvije operacije uklanjuju nepotrebne informacije o bojama sa slike što omogućuje bržu i jednostavniju obradu i analizu slike. Naravno to ima smisla samo ako nam informacije o tim bojama nisu potrebne.

Osim nepotrebnih boja, bukom se može smatrati i nasumična razlika u vrijednosti između susjednih piksela. U digitalnoj fotografiji buka je praktički neizbjegljiva i može se definirati kao nasumična varijacija signala u slici. Ta varijacija je reprezentirana različitim nijansama boja susjednih piksela. Buka se može smanjiti manipulacijom parametara digitalne kamere, ali ne može se kompletno ukloniti [19].

Takve informacije se zbog malih razlika u boji piksela većinom mogu smatrati nevažnim, a loše utječu na daljnju obradu slike. Zbog toga postoje razne tehnike za filtriranje slike. Najpopularnije tehnike za filtriranje buke su [20]:

- Gaussov filter (*eng. gaussian filter*)
- Srednji filter (*eng. mean filter*)
- Medijanski filter (*eng. median filter*)
- Bilateralni filter (*eng. bilateral filter*)

Svaki tip filtera ima svoju upotrebu i koristan je za određeni tip buke. Buka u digitalnim slikama je jako opširna tema koje se neće detaljnije objašnjavati u ovom radu.



Slika 5: Primjer medijanskog filtera [21]

Na priloženoj slici je prikazan proces korištenja medijanskog filtera koji naspram matrice susjednih vrijednosti određuje vrijednost piksela koristeći medijan svih vrijednosti.

Važno je napomenuti da su pojmovi poput oštine i mutnoće slike usko povezani s bukom. Jako često mutna slika sadrži manje buke, dok oštra slika može sadržavati više buke. Zbog toga je potrebno naći ravnotežu pri uklanjanju buke. Ako uklonimo previše buke, slika postaje mutna i gubi se previše informacija, obrnuto vrijedi ako uklonimo preveliko buke.

Uklanjanje buke jedan je od najvažnijih koraka jer se nakon njega u dalnjim fazama ne analiziraju nepotrebne i neželjene informacije. Moglo bi se reći da je to de facto obavezan korak preprocesiranja, budući da se izrazito često koristi.



Slika 6: Usporedba procesirane slike bez i s preprocesiranjem (vlastita izrada)

Na priloženoj slici moguće je vidjeti kako je znatno manje nevažnih informacija na desnoj binarnoj slici jer je preprocesiranjem uklonjena buka, čime su dobiveni puno čišći podaci koji se mogu koristiti za daljnju obradu.

Postoji još razne metode poboljšavanja slike poput povećavanja kontrasta, ali neće biti potrebne u ovom radu. Također u nekim literaturama se spominju tehnike poput smanjenja buke i raznih drugih transformacija u kontekstu restauracije slike. Ali u ovom radu će se o tome govoriti u kontekstu poboljšanja slike.

Druge važne metode poboljšavanja slike su razne geometrijske transformacije slike po-

put translacije, rotacije i skaliranja. Sve te osnovne geometrijske operacije koriste se kod ispravljanja perspektive. Baš zbog toga je ispravak perspektive puno zanimljivija operacija za ovaj rad [22].

Ispravljanje perspektive omogućuje da slika prikazuje kao da je slikana u nekoj drugoj ravnini u prostoru. To omogućuje da se uklone neželjene značajke pribavljene slike poput kuta kamere naspram subjekta fotografije, u ovom slučaju papira s matricom odgovora.

Korekcija perspektive mapira točku  $\mathbf{p} = [x, y, 1]^T$  u originalnoj slici (slikanoj iz nekog neželjenog kuta)  $\mathbf{p}' = [x', y', w']^T$  u sliku ispravljene perspektive s pomoću matrice  $H$  [23].

$$\mathbf{p}' = H \cdot \mathbf{p}$$

Gdje  $H$  predstavlja  $3 \times 3$  homografsku matricu:

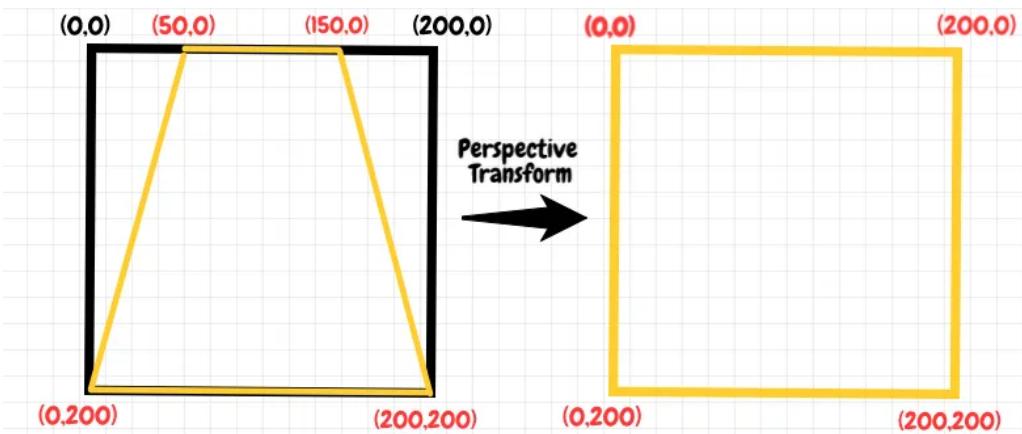
$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

Transformirane koordinate se računaju na sljedeći način:

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Da bi te koordinate bile korisne potrebno ih je normalizirati s  $w$ :

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \quad y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}$$



Slika 7: Prikaz procesa ispravka perspektive subjekta na slici [23]

Na priloženoj slici prikazano je kako se matrica transformacije koristi da bi se perspektiva ispravila. U praksi to funkcioniра tako da se matrica transformacije računa naspram 4

izvorne i 4 odredišne točke. Te se zatim prema formuli množi sa svakom izvornom točkom da bi se izračunale odredišne točke.

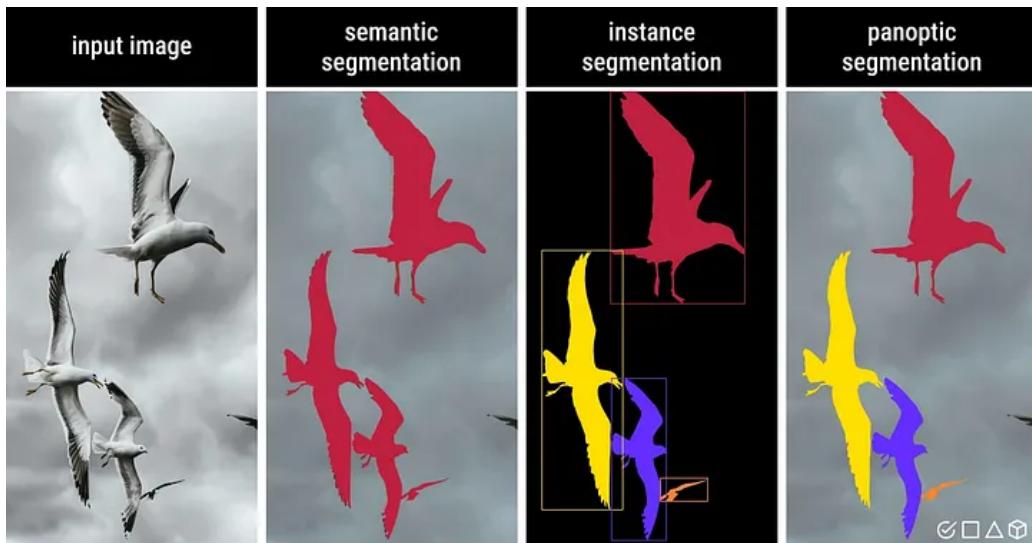
### 3.4.2. Procesiranje slike

Nakon što je faza preprocesiranja pretvorila sliku u korisniji format, kreće glavna faza obrade to jest procesiranje slike. Glavne tehnike kod procesiranja slike su segmentacija, detekcija objekata, morfološka obrada te reprezentacija i opis.

#### 3.4.2.1. Segmentacija slike

Segmentacija slike je proces grupiranja piksela prema nekim obilježjima. Za razliku od običnih klasifikatora poput neuronskih mreža, segmentacija nudi informacije gdje se točno na slici nalazi objekt od interesa i koji su sve pikseli dio tog objekta. Isto kao u kod klasifikatora potrebno je definirati moguće klase kojima piksel može pripadati. Definiranje tih klasa može se učiniti na tri načina, prema kojima dijelimo tipove segmentacije na [24]:

- semantička segmentacija (*eng. semantic segmentation*)
- segmentacija instance (*eng. instance segmentation*)
- panoptička segmentacija (*eng. panoptic segmentation*)



Slika 8: Usporedba tipova segmentacije [24]

Na priloženoj slici definirane su dvije klase: nebo i ptice. Semantička segmentacija je sve piksele ptica klasificirala kao jednu klasu, što znači da ne znamo točno kojoj ptici pripada pojedini piksel. Da bi to riješili postoji segmentacija prema instanci što odvaja svaku pticu u zasebnu klasu što je bolje od semantičke, ali znači da više ne znamo koji sve pikseli pripadaju roditeljskoj klasi ptica. Da bi to riješili postoji panoptička segmentacija koja kombinira ta dva

pristupa i pruža informacije o klasi piksela i kojoj instanci pripada. Tip segmentacije ovisi o specifičnostima problema koji se pokušava riješiti [24].

### 3.4.2.2. Morfološka obrada

Morfološka obrada je skup operacija koje procesiraju sliku. Omogućuje pronalazak elemenata na slici sličnih definiranom obliku. Usporedbu oblika vrši tako da traži grupu piksela koji odgovaraju tom obliku [25].

Morfološka obrada se u nekim segmentima preklapa s segmentacijom slike, ali ne radi se o istim operacijama. Morfološke operacije se često koriste nakon segmentacije da uklone nedostatke iz detektiranih segmenata [25].

Dvije najvažnije morfološke operacije su erozija i dilatacija. Erozija funkcioniра na način da uklanja piksele s ruba objekta. Vrlo je korisna u slučajevima kada je potrebno odvojiti dva objekta određenog oblika [25].



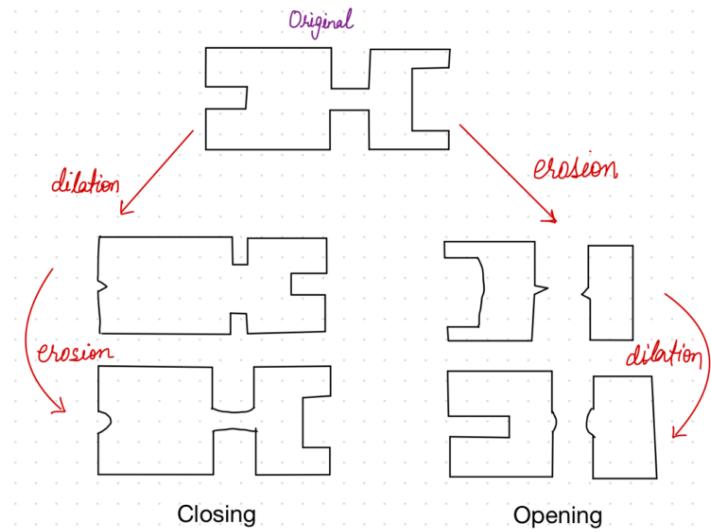
Slika 9: Prikaz erozije kao morfološke operacije [25]

Za razliku od erozije, dilatacija dodaje piksele oko ruba objekta. Time omogućuje da se popune praznine između objekata.



Slika 10: Prikaz dilatacije kao morfološke operacije [25]

Najčešće se te dvije operacije ne koriste zasebno već zajedno kao kompozitna morfološka operacija. Ako prvo provedemo eroziju, zatim dilataciju, radi se o otvaranju, dok obrnuti redoslijed operacija čini zatvaranje. [25].



Slika 11: Prikaz kompozitnih morfoloških operacija [25]

### 3.4.2.3. Reprezentacija i opis

Nakon što je obavljena segmentacija i po potrebi morfološka obrada većinom slijedi reprezentacija i opis segmentiranih dijelova. Cilj reprezentacije i opisa je reprezentirati piksele tako da budu korisniji za daljnju obradu [26].

Dva glavna tipa reprezentacije segmentiranih objekata, ovisno o tome je li fokus na [26]:

- vanjskim karakteristikama, (kontura ili rub objekta)
- unutarnjim karakteristikama (pikseli koji čine objekt)

Reprezentacija prema vanjskim karakteristikama se često koristi kad nam je za obradu važan sam oblik, dok se unutarnje karakteristike koriste kada je potrebna informacija o boji i teksturi objekta. Iz dobivenih reprezentacija je moguće izvući razne opisne vrijednosti (*eng. descriptors*) koje su korisne kod daljne obrade. Važno je napomenuti da je generalno pravilo da bi opisne vrijednosti trebale biti imune na promjenu skaliranja, translacije i rotacije [26].

Neke od korisnik opisnih vrijednosti su [26]:

- Dužina konture objekta
- Promjer konture objekta
- Zaobljenost
- Okvirna kutija
- Površina objekta

- Sličnost željenom obliku
- Odstupanje od simetrije
- Koeficijent istezanja

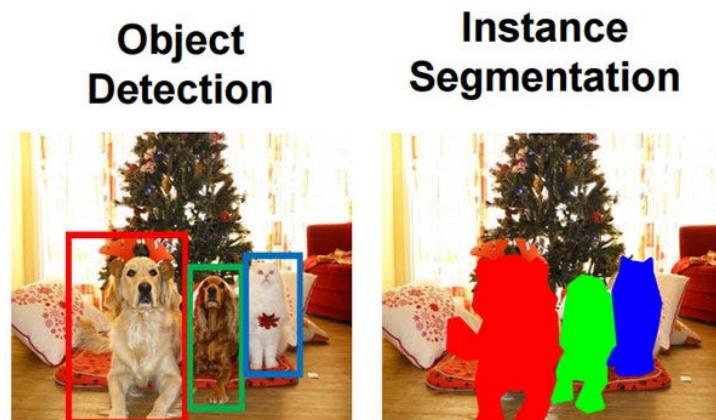
#### 3.4.2.4. Detekcija i prepoznavanje objekata

Detekcija i prepoznavanje objekata je jedna od ključnih tehnika procesiranja slika na tradicionalan način, ali i s pomoću neuronskih mreža. Bazirano je na principu pronađaska objekta i crtanja okvirne kutije oko tog objekta [27], [28].

To mu je ujedno i glavna razlika naspram segmentacije slike, segmentacija dodjeljuje klasu svakom pikselu koji odgovara onome što se traži, dok detekcija objekta to ne radi već samo locira objekt određene klase na slici [27], [28].

Ponekad je samo informacija o lokaciji objekta dovoljna, u slučaju da nije segmentacija nudi puno više informacija o samom objektu.

Da bi detektiranje objekata bilo uspješno često se koriste konture. Konture su ništa drugo nego krivulje koje opisuju rubove objekata, zbog toga su jako korisne kod detektiranja oblika. Također se ponekad koriste i kod segmentiranja.

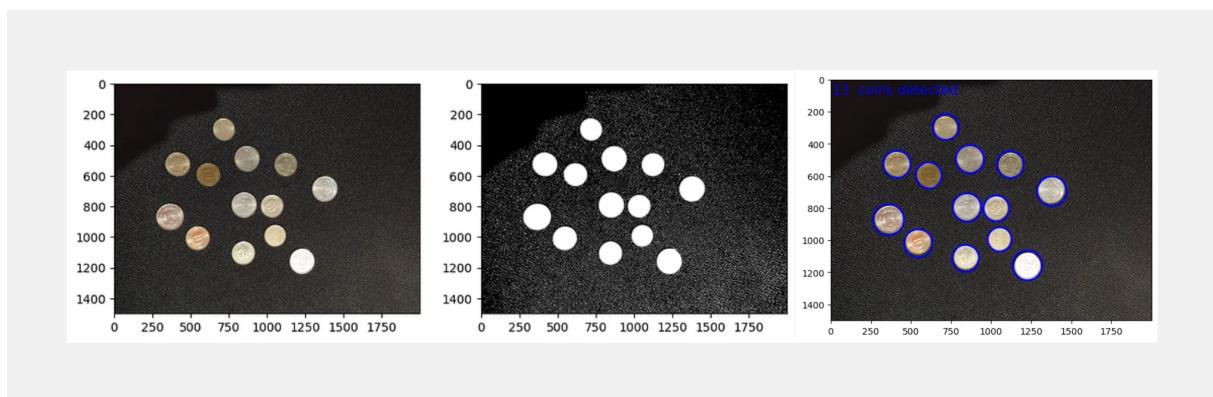


Slika 12: Usporedba segmentacije i detekcije objekta [28]

Da bi se moglo pronaći konture prvo je potrebno sliku pretvoriti u binarnu sliku i zatim provesti operaciju detektiranja kontura. Konture se detektiraju tako da se traži razlika između kontrasta piksela na binarnoj slici i time se utvrđuje ujedno i granica između objekata. Zatim se primjenjuje jedna od metoda za detekciju objekta nad tim konturama. Te metode zapravo traže konture koje odgovaraju traženom obliku s definiranim tolerancijama za oblik, veličinu i slično [29].

Budući da se u ovom radu koristi pristup temeljen na tradicionalnoj obradi slike, od posebnog su interesa sljedeće metode koje omogućuju detekciju i analizu geometrijskih oblika koristeći konture:

- **Aproksimacija kontura u poligone** - koristi se za pojednostavljenje kontura i prepoznavanje osnovnih geometrijskih oblika.
- **Izračun minimalnog pravokutnika oko konture** - omogućuje određivanje najmanjeg pravokutnika koji obuhvaća određeni objekt, što je korisno pri detekciji pravokutnih oblika.
- **Prilagodba elipse konturi** - koristi se za identifikaciju objekata eliptičnog oblika.
- **Detekcija ravnih linija Houghovom transformacijom** - omogućuje otkrivanje linijskih struktura unutar slike.
- **Detekcija kružnica Houghovom transformacijom** - primjenjuje se za prepoznavanje kružnih objekata.
- **Pronalaženje pravokutnog okvira oko konture (bounding box)** - omogućuje okvirno lociranje objekta unutar slike.
- **Detekcija kontura s hijerarhijskim odnosima** - omogućuje razlikovanje vanjskih i unutarnjih kontura, što je korisno pri analizi složenijih objekata.



Slika 13: Prikaz rezultata detekcije krugova pomoću kontura [28]

Na priloženoj slici je jasno prikazano kako se izvorna slika pretvara u binarnu sliku i zatim se s pomoću kontura i metode za detektiranje oblika naspram kontura pronalazi lokacija kruga.

### 3.4.3. Postprocesiranje slike

Nakon što je glavna faza obrade slike završena kreću aktivnosti postprocesiranja. Prijašnja faza je izvukla korisne podatke iz slike i pripremila ih u korisnom formatu. Cilj postprocesiranje je obrada tih podataka tako da se dođe do konačnog rješenja. Osim obrade podataka, postprocesiranje je također povezano s aktivnostima poput vizualizacije rezultata, kompresije i slično.

### **3.4.3.1. Obrada izvučenih podataka**

Prije nego što se krene u konačnu obradu izvučenih podataka potrebno ih je pročistiti. Načini pročišćavanja podataka ovise o specifičnostima problema, ali neke od važnih metoda za ovaj rad su:

- Filtriranje po veličini objekta
- Filtriranje po obliku objekta
- Filtriranje po boji objekta
- Filtriranje po hijerarhijskoj odnosu objekta s ostalim objektima

Spomenute metode bazirane su na usporedbi opisnih vrijednosti iz faze reprezentacije i opisa. Nakon što su podaci pročišćeni kreće konačna faza obrade. Način konačne obrade ovisi o specifičnostima problema, ali često se koriste baze znanja.

Baze znanja omogućuju usporedbu traženih podataka s pročišćenim podacima koje omogućuje rješenje specifičnog problema. U ovom slučaju baza znanja bi bila lista odgovora za svako pitanje podijeljenih u dvije klase koje predstavljaju točne i netočne odgovore [15].

### **3.4.3.2. Vizualizacija rezultata**

Nakon što su podaci obrađeni i utvrđen je rezultat jako često ga je potrebno vizualizirati. Vizualizacija nudi uvid u njihovu važnost i omogućuje da se jednostavnije analiziraju i izvode zaključci iz rezultata. Tip vizualizacije ovisi o domeni problema koji se rješava. Može biti u raznim oblicima poput PDF izvješća, vizualizacije rezultata nad originalnom slikom i slično [30].

Kod vizualiziranja rezultata važno je uzeti u obzir tip podataka koji se vizualizira, publiku za koju je vizualizacija namijenjena [30].

### **3.4.3.3. Kompresija slike**

U slučaju potrebe za smanjenjem veličine slike, često se koristi kompresija. Komprezija nastoji smanjiti veličinu slike na način da pritom zadrži kvalitetu na zadovoljavajućoj razini. Često se primjenjuje kada je potrebno smanjiti veličinu slike pri prijenosu preko mreže, spremanju u memoriju ili za ubrzanje obrade [15].

Kompresija slike dijeli se na dva glavna tipa [31]:

- Kompresija s gubitkom podataka (*eng. lossy*)
- Kompresija bez gubitka podataka (*eng. lossless*)

Oba tipa smanjuju veličinu slike, ali se razlikuju u načinu na koji to postižu. Prvi pristup uklanja određene informacije iz slike kako bi smanjio njezinu veličinu (*lossy*), dok drugi pristup

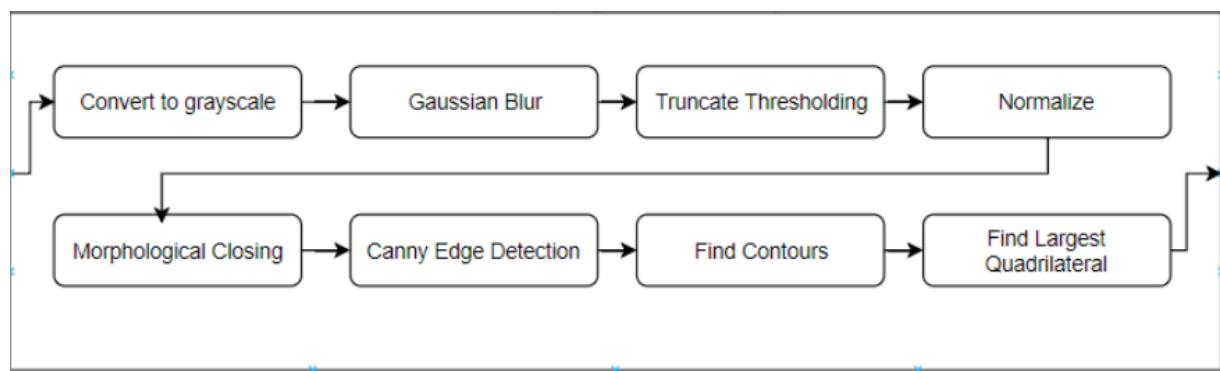
efikasnije zapisuje podatke bez gubitka informacija (lossless). Važno je napomenuti da je kod kompresije uvijek potrebno balansirati veličinu slike i kvalitetu. Što je slika više komprimirana, to je izraženija distorzija slike. Glavna razlika između ovih dvaju pristupa je u tome može li se originalna slika potpuno rekonstruirati iz komprimirane slike [31].

### 3.5. Analiza postojećih rješenja i pristupa problemu

Prije praktičnog dijela rada važno je analizirati postojeća rješenja na tržištu. Ova analiza fokusirat će se samo na detekciju odgovora na pismenim ispitima koristeći matricu odgovora to jest detekciju OMR tehnikom (*eng. optical mark recognition*). Pod OMR spadaju sve tehnike koje su do sada obrađene. Fokus ove analize je samo na rješenjima visoke razine točnosti.

Osim korištenja računalnog vida uz OMR sve češći su pristupi koji koriste elemente dubokog učenja, samostalno ili u kombinaciji s tradicionalnim računalnim vidom [32].

Analizirana su rješenja OMRChecker i OpenMCR, radi se o rješenjima izrazito visoke točnosti koji tu točnost zadržavaju i kada je slika uslikana u izrazito izazovnim uvjetima. Također oba rješenja podržavaju određeni stupanj konfiguracije liste s odgovorima [32].



Slika 14: Prikaz dijela toka procesiranja slike kod OMRChecker-a [33]

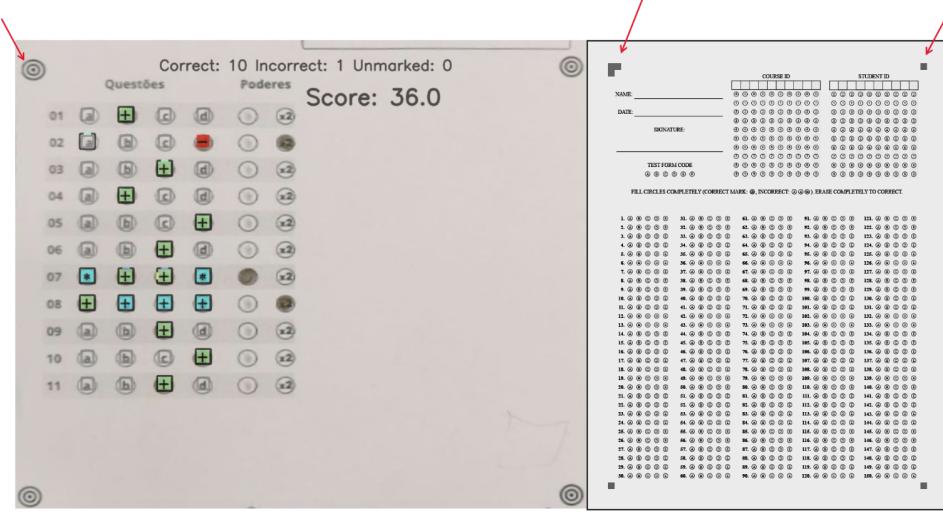
S priložene slike vidljivo je da OMRChecker koristi veliku većinu metoda i tehnika koje su do sada obrađene u ovom radu, za OpenOMR je sitacija također dosta slična.

Osim samih značajki sustava važno je znati kako su se ti sustavi nosili sa problemima pri detekciji odgovora spomenutih u prijašnjim poglavljima. Jedan od glavnih problema je kvaliteta pribavljenе fotografije. To uključuje loše uvjete osvjetljenja, udaljenost uređaja od papira, kut snimanja i slične čimbenike.

Rješenja poput OpenMCR rješila su problem kuta snimanja i rotacije papira tako da su na matricu odgovora postavljena posebne oznake pomoću kojih se precizno detektiraju kutovi matrice i određuje orientacija papira.

Na priloženoj slici prikazana su te posebne oznake koja olakšavaju detekciju kutova papira. Na lijevoj slici prikazana je matrica odgovora OMRChecker sustava, koja koristi četiri identične oznake za detekciju kutova papira. Na desnoj slici prikazana je matrica odgovora OpenMCR sustava, koja koristi tri identične oznake i jednu unikatnu. Ta različita oznaka omogućuje ne samo detekciju kutova, već i određivanje orientacije papira.

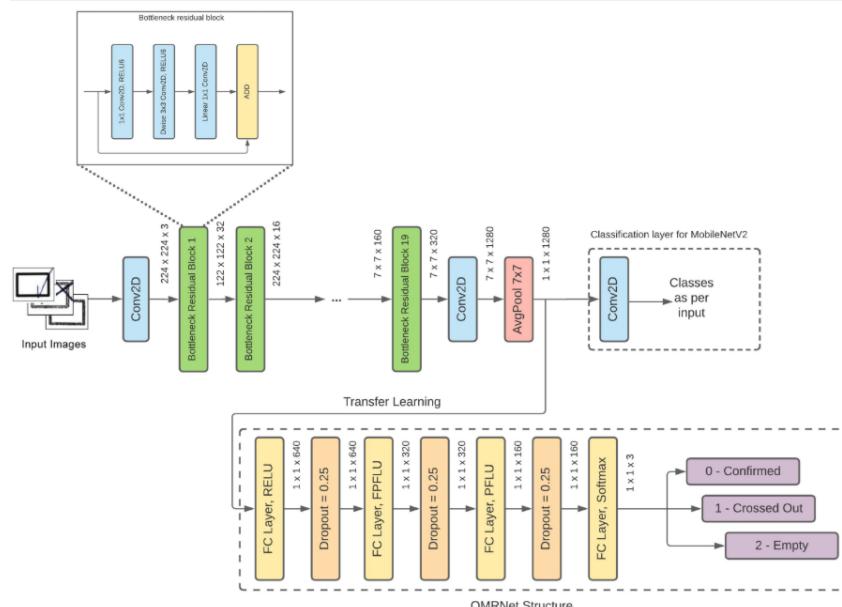
To znači da OpenMCR nudi kvalitetnije rješenje barem što se tiče korekcije kuta snimanja i rotacije papira u odnosu na OMRChecker.



Slika 15: Prikaz pomoćnih oznaka za detektiranje kutova matrice odgovora i orijentacije [33] [34]

Osim problema s kvalitetom pribavljene slike, postoje i problemi s neispravnim označavanjem područja za odgovor. Taj je problem teško rješiti korištenjem tradicionalne obrade slike, te stoga ima više smisla koristiti strojno učenje, odnosno određeni tip klasifikatora [35].

OMRNet je jedno od takvih rješenja. Ono nudi mogućnost klasifikacije područja odgovora kao označeno, neoznačeno i prekriženo. Zapravo se ne radi o cijelovitom sustavu za ocjenjivanje ispita, već o sustavu koji pruža REST API za klasifikaciju područja odgovora, te se može integrirati s drugim sustavima za detekciju odgovora i ocjenjivanje ispita [35].



Slika 16: Prikaz strukture OMRNet klasifikatora [35]

Također, važno je napomenuti da ova konvolucijska neuronska mreža koristi metodu prijenosa znanja (eng. transfer learning) s već prethodno treniranog modela MobileNet V2 [35].

Cijela svrha korištenja OMRNet-a ili neke druge implementacije dubokog učenja za klasifikaciju područja odgovora jest rješavanje problema koje OMR sustavi temeljeni isključivo na tradicionalnoj obradi slike imaju [35]:

- Teško je razaznati radi li se o prekriženom odgovoru ili o označenom odgovoru
- Problem definiranja praga koji određuje je li odgovor označen ili nije

Važno je napomenuti da su sva analizirana rješenja u određenoj mjeri ograničena kada je riječ o fleksibilnosti predloška matrice odgovora. Naime, postojeća rješenja očekuju da se na papiru nalaze određene strukture, poput oznaka za kutove i orijentaciju, specifičan broj pitanja i odgovora i slično. Sama fleksibilnost rješenja ovisi o tome je li sustav dizajniran da strogo slijedi zadani predložak ili je već u samom dizajnu predviđena određena razina prilagodljivosti.

Također, sva rješenja variraju u razini pouzdanosti i konzistentnosti u ekstremnim uvjetima rada. Pod time se podrazumijevaju visoka razina šuma, loši uvjeti osvjetljenja, ekstremni kutovi fotografiranja i slično. Međutim, postavlja se pitanje je li uopće nužno da sustav pouzdano funkcioniра u takvim ekstremnim uvjetima. Na primjer nije jednaka razina pouzdanosti potrebna sustavu za ocjenjivanje ispita i sustavu za čitanje registarskih tablica na parkiralištu. Glavni razlog za to je činjenica da ocjenjivač može u velikoj mjeri kontrolirati uvjete i način pribavljanja fotografije, dok na parkiralištu to najčešće nije moguće.

Osim navedenih ograničenja, analizirana rješenja imaju i niz prednosti. Sva rješenja postižu izrazito visoku razinu točnosti koja je otprilike 99,5%. Važno je istaknuti da za postizanje te razine preciznosti nisu potrebni značajni računalni ni vremenski resursi. Takvi sustavi također značajno skraćuju sam proces ocjenjivanja ispita u usporedbi s čovjekom. Iako ovi sustavi nisu savršeni, riječ je o zreloj i relativno jeftinoj tehnologiji koja pokazuje znatno nižu stopu pogrešaka u odnosu na čovjeka, pogotovo kada se uzme u obzir vrijeme potrebno za ocjenjivanje.

Za kraj, važno je napomenuti da je takve sustave teško međusobno usporedjivati, budući da se implementacije često razlikuju samo po korištenim metodama, dok su same faze obrade u većini slučajeva jednake i već su obrađene u prijašnjim poglavljima. Dobar primjer toga je otklanjanje šuma, koje je praktički neizostavan korak u svim sustavima, razlika je jedino u konkretnoj metodi koja se koristi za tu svrhu.

Ipak, postoje i inovativna rješenja koja pristupaju postojećim problemima na kreativan način. Dobar primjer za to je OMRNet klasifikator, koji primjenom metode dubokog učenja nudi novu perspektivu u detekciji označenih odgovora. Kombinacijom različitih aspekata postojećih rješenja moguće je dizajnirati kvalitetan i precizan sustav za detekciju i ocjenjivanje pismenih ispita zatvorenog tipa.

## 4. Definiranje zahtjeva

Kroz daljnji dio rada fokus će biti na izradi i prikazu komponente otvorenog koda za detekciju odgovora na pismenim testovima. Prikazat će se svi važni koraci i dizajnerske odluke koje utječu na konačni dizajn i funkcionalnost rješenja. Izrađena komponenta će također biti ugrađena u jednostavnu stolnu aplikaciju da bi se olakšalo razvoj i vizualizirao sam rad komponente.

### 4.1. Opis rješenja

Kao što je već spomenuto cilj je izraditi komponentu otvorenog koda za detekciju odgovora na pismenim ispitima. Fokus će biti na detektiranju odgovora s matrice odgovora. Prednost toga pristupa je to što nije potrebno slikati više stranica da bi se ocijenio jedan ispit, već je dovoljno ocijeniti samo matricu odgovora, to jest obraditi samo jednu stranicu.

Taj pristup detekciji odgovora je puno jednostavniji problem za riješavanje nego detekcija na standardnim ispitima. Jednostavniji je zbog toga što matrica može imati unaprijed definiranu i standardiziranu strukturu to jest predložak. Unatoč tome matrica limitira rješenje samo na pitanja s višestrukim ili jednostrukim odgovorima. No to nije problem za ovaj rad pošto drugi tipovi pitanja nisu u opsegu ovog rada.

Jedan od glavnih ciljeva ovog rješenja je da konzistentno pruža pouzdane rezultate, čak i ako uvjeti u kojima je slika uslika nisu idealni. Ta pouzdanost se treba postići za predloške koji mogu sadržavati varijabilan broj pitanja i odgovora. Da bi izrada predložaka bila jednostavnija, rješenje bi također trebalo biti sposobno prema određenim parametrima generirati predložak matrice odgovora u obliku PDF-a.

Također da bi se ubrzao rad s aplikacijom ideja je da korisnik ne unosi ručno podatke o točnim odgovorima za sva pitanja, već jednostavno slika ispravno riješen ispit te da komponenta sama izvuče te informacije iz fotografije. Da bi se dodatno ubrzao proces ocjenjivanja sustav treba biti dizajniran na način da detektirane odgovore usporedi s ispravnim odgovorima i ujedno prema korisnički definiranoj skali ocijeni ispit. Ocijenjeni ispit mora biti vizualiziran kako bi ocjenjivač mogao uvidjeti je li došlo do greške kod ocjenjivanja.

U svrhu bržeg razvoja, pronalaska grešaka i slično sustav bi trebao imati mogućnost vizualiziranja svakog koraka obrade od samog unosa slike pa sve do rješenja. Ta opcija bi se trebala moći uključiti ili isključiti po potrebi.

Budući da se radi o komponenti otvorenog koda želja je da se ostavi mogućnost zamjene važnih biblioteka koje će sustav koristiti, poput one za računalni vid. Time će se omogućiti da netko po potrebi napiše svoju implementaciju i zamijeni zadalu EmguCv biblioteku.

## 4.2. Funkcionalni zahtjevi

<b>Identifikator</b>	FZ-01
<b>Zahtjev</b>	Sustav mora detektirati odgovore s matrice odgovora.
<b>Opis</b>	Detekcija se temelji isključivo na obradi jedne stranice to jest matrice odgovora. Odgovori se trebaju detektirati, klasificirati prema tome jesu li označeni i zatim ih pridužiti odgovarajućem pitanju.
<b>Način provjere</b>	Prilikom unosa slike matrice odgovora, sustav treba pravilno označiti sve odgovore na slici i označitiji ih odgovarajućom bojom ovisno o klasi kojoj pripadaju.
<b>Prioritet [1–5]</b>	5
<b>Identifikator</b>	FZ-02
<b>Zahtjev</b>	Sustav mora omogućiti generiranje PDF predložaka za matricu odgovora.
<b>Opis</b>	Korisnici trebaju moći jednostavno izraditi matricu s unaprijed definiranim brojem pitanja i odgovora.
<b>Način provjere</b>	Iz sustava treba biti moguće izvesti generirani predložak u PDF formatu, parametri za unos su broj pitanja i broj odgovora po pitanju.
<b>Prioritet [1–5]</b>	3
<b>Identifikator</b>	FZ-03
<b>Zahtjev</b>	Sustav mora omogućiti automatsko prepoznavanje točnih odgovora iz slike ispravno riješenog ispita.
<b>Opis</b>	Korisnik ne mora ručno unositi odgovore; slika ispravnog rješenja bit će dovoljna za generiranje ključa.
<b>Način provjere</b>	Učitavanjem slike ispravno riješenog ispita sustav mora automatski prepoznati sve točne odgovore.
<b>Prioritet [1–5]</b>	4
<b>Identifikator</b>	FZ-04
<b>Zahtjev</b>	Sustav mora usporediti odgovore s točnim rješenjem i izračunati ocjenu prema skali koju je korisnik definirao.
<b>Opis</b>	Automatsko ocjenjivanje ubrzava proces ispravljanja i smanjuje mogućnost pogreške.
<b>Način provjere</b>	Nakon što se ispiti obrade, mora biti prikazana ocjena za svaki ispit u skladu sa skalom.
<b>Prioritet [1–5]</b>	4

<b>Identifikator</b>	FZ-05
<b>Zahtjev</b>	Sustav mora omogućiti vizualizaciju rezultata ocjenjivanja za svakog korisnika.
<b>Opis</b>	Omogućava ljudsku kontrolu rezultata i ispravnost prepoznavanja i ocjenjivanja.
<b>Način provjere</b>	Vizualna prezentacija ispita s označenim točnim i netočnim odgovorima mora biti dostupna.
<b>Prioritet [1–5]</b>	3
<b>Identifikator</b>	FZ-06
<b>Zahtjev</b>	Sustav mora omogućiti prikaz svih koraka obrade slike.
<b>Opis</b>	Korisnicima i developerima ova funkcionalnost je korisna za praćenje grešaka i ponašanja sustava. Slike se trebaju prikazati tijekom obrade i spremiti na disk računala.
<b>Način provjere</b>	Funkcionalnost prikazuje korake i sprema ih na disk ako je uključena, inače ne.
<b>Prioritet [1–5]</b>	5
<b>Identifikator</b>	FZ-07
<b>Zahtjev</b>	Sustav mora podržavati pitanja s jednostrukim i višestrukim odgovorima.
<b>Opis</b>	Korisnici trebaju imati mogućnost definiranja pitanja koja imaju jedan točan odgovor (jednostruki izbor) ili više točnih odgovora (višestruki izbor). Sustav mora pravilno detektirati i ocijeniti oba tipa pitanja.
<b>Način provjere</b>	Prilikom testiranja, sustav mora ispravno ocijeniti ispite koji sadrže oba tipa pitanja.
<b>Prioritet [1–5]</b>	4

Tablica 2: Funkcionalni zahtjevi

### 4.3. Nefunkcionalni zahtjevi

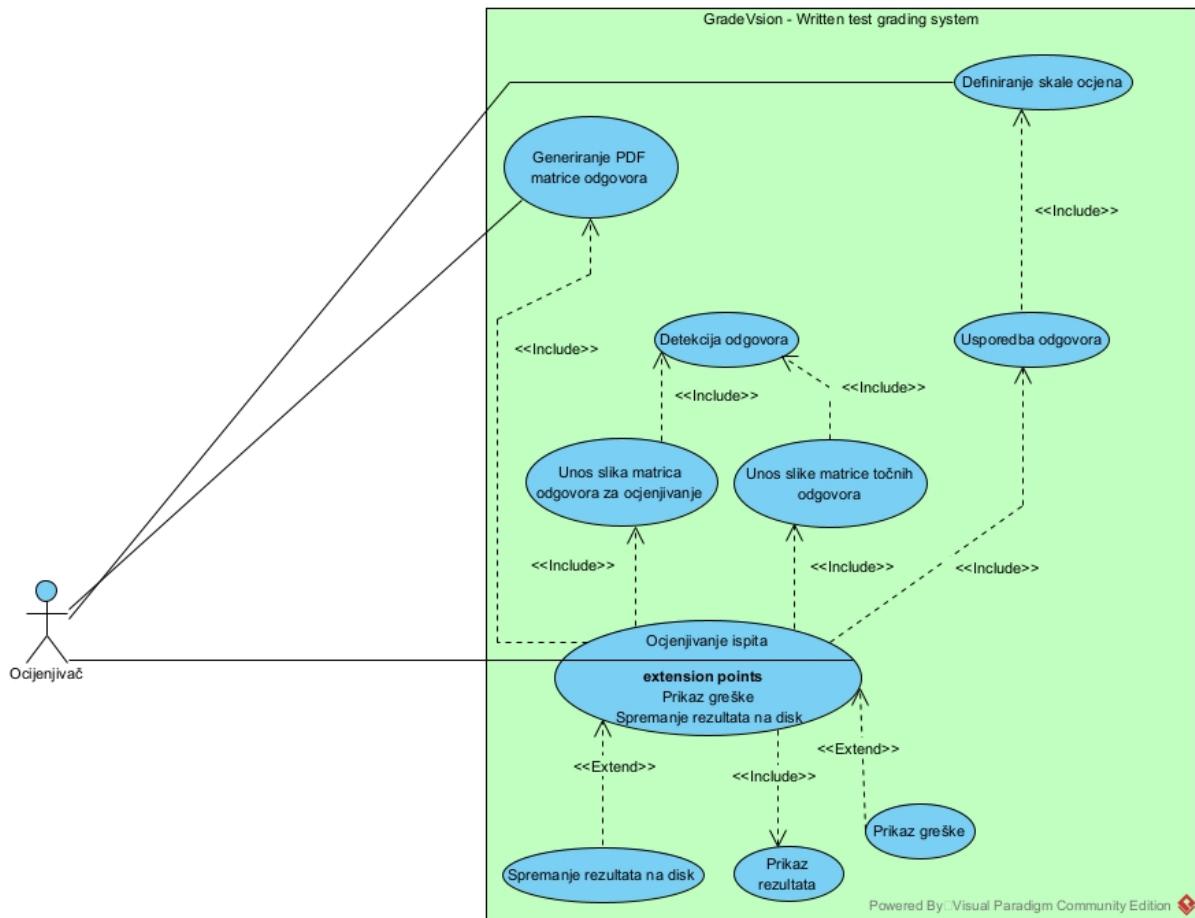
<b>Identifikator</b>	NFZ-01
<b>Zahtjev</b>	Sustav mora pružati pouzdane rezultate čak i u slučaju loših uvjeta pri snimanju (npr. sjenke, loš kontrast, zakrenutost).
<b>Opis</b>	U praksi slike ispita neće uvijek biti idealne kvalitete, sustav mora biti sposoban obraditi sliku i u takvim uvjetima.
<b>Način provjere</b>	Sustav se testira na uzorku slika s različitim uvjetima osvjetljenja, orientacije i kvalitete.
<b>Prioritet [1–5]</b>	3

<b>Identifikator</b>	NFZ-02
<b>Zahtjev</b>	Sustav mora omogućiti jednostavnu zamjenu vanjskih biblioteka poput one za računalni vid.
<b>Opis</b>	Komponenta pruža sučelje koje korisnici mogu naslijediti i napisati svoju implementaciju i time zamijeniti biblioteku za CV.
<b>Prioritet [1–5]</b>	2
<b>Identifikator</b>	NFZ-03
<b>Zahtjev</b>	Sustav mora moći detektirati odgovore na predlošku s varijabilnim brojem pitanja i odgovora
<b>Opis</b>	Korisnici će moći generirati predložak s različitim brojem pitanja i odgovora, sustav ih treba sve ispravno detektirati.
<b>Način provjere</b>	Testirati s raznim kombinacijama matrica odgovora sposobnost detekcije kada broj pitanja i odgovora nije fiksan, već varijabilan.
<b>Prioritet [1–5]</b>	2
<b>Identifikator</b>	NFZ-04
<b>Zahtjev</b>	Sustav mora pronaći odgovore na slici unutar 500 milisekundi.
<b>Opis</b>	Nakon što korisnik učita sliku i pritisne gumb za detekciju odgovora, sustav ima najviše 500 milisekundi za obradu slike i pronalazak odgovora. Ovo ograničenje vrijedi za slike koje ne prelaze 2K rezoluciju te se odnosi isključivo na rad izvan načina za otklanjanje pogrešaka. Vrijeme učitavanja i spremanja slika (I/O operacije) nije uključeno u navedeno vremensko ograničenje.
<b>Način provjere</b>	Izmjeriti prosječno vrijeme potrebno za detekciju odgovora sa slike u standardnim uvjetima.
<b>Prioritet [1–5]</b>	1

Tablica 3: Nefunkcionalni zahtjevi

#### 4.4. Slučajevi korištenja

Kroz dijagram slučajeva korištenja moguće je vidjeti koji su korisnici sustava te koja je razina njihove interakcije s određenim funkcijama i dijelovima sustava. Važno je napomenuti da se radi o prikazu visoke razine, čiji elementi ne moraju nužno odgovarati programskom kodu jedan na jedan. Na konkretnom primjeru ovog sustava može se uočiti da su ocjenjivači jedini korisnici sustava. Oni imaju mogućnost iniciranja više različitih akcija unutar sustava.



Slika 17: Prikaz Use-case diagrama (Vlastita izrada)

#### 4.4.1. Slučaj korištenja: Ocjenjivanje ispita

##### 4.4.1.1. Korisnici

Kao što je već spomenuto, ocjenjivači su jedini korisnici ovog sustava. Oni su u koначnici odgovorni za sam proces ocjenjivanja ispita, a GradeVision softversko rješenje im služi kao alat koji im u tom procesu značajno pomaže. Pomaže im tako što ubrzava sam proces ocjenjivanja, što im omogućuje da više vremena posvete drugim obvezama, poput podizanja kvalitete nastave i sl.

##### 4.4.1.2. Glavni scenarij

Glavni scenarij u ovom sustavu je ocjenjivanje ispita. Ocjjenivač pokreće tu aktivnost, koja zatim uključuje ostale dijelove sustava o kojima ovisi ispravan rad. To su druge aktivnosti poput generiranja PDF matrice odgovora, unosa slike i usporedbe odgovora.

#### **4.4.1.3. Uključeni slučaj uporabe: Generiranje PDF matrice odgovora**

Sustav ima mogućnost ocjenjivanja samo onih ispita koji imaju matrice odgovora u određenom formatu, stoga se prije ocjenjivanja ta matrica mora generirati. Taj format je donekle fleksibilan što se tiče broja pitanja i odgovora, ali osnovna struktura mora biti ista. Zbog toga ocjenjivač prije samog ocjenjivanja mora pokrenuti aktivnost "Generiranje PDF matrice odgovora" i definirati broj pitanja i odgovora kako bi kasnije mogao uspješno ocijeniti te matrice. Ako se ovaj korak preskoči i u sustav se učita matrica nekompatibilnog formata, sustav neće raditi.

#### **4.4.1.4. Uključeni slučaj uporabe: Unos slike matrice točnih odgovora**

Također, prije ocjenjivanja sustav mora znati koji su točni odgovori kako bi ih kasnije mogao usporediti. Stoga ocjenjivač mora učitati sliku matrice točnih odgovora čime se pokreće aktivnost "Unos slike matrice točnih odgovora".

#### **4.4.1.5. Uključeni slučaj uporabe: Unos slike matrica odgovora za ocjenjivanje**

Nakon što sustav ima učitanu sliku matrice točnih odgovora, potrebno je učitati slike matrica odgovora koje treba ocijeniti, jer bez tog koraka sustav ne zna što je potrebno ocijeniti. Učitavanjem tih slika pokreće se aktivnost "Unos slike matrica odgovora za ocjenjivanje".

#### **4.4.1.6. Uključeni slučaj uporabe: Detekcija odgovora**

Obje aktivnosti učitavanja matrica odgovora ovise o aktivnosti "Detekcija odgovora", koja je ujedno jedna od najvažnijih aktivnosti koje sustav provodi, budući da se tu nalazi cijela logika za obradu slika i izdvajanje podataka o označenim odgovorima.

#### **4.4.1.7. Uključeni slučaj uporabe: Usaporedba odgovora**

Nakon što su odgovori s učitanih matrica odgovora detektirani, pokreće se aktivnost "Usaporedba odgovora", koja uspoređuje odgovore s matrica za ocjenjivanje s točnim odgovorima iz matrice točnih odgovora. Ta aktivnost također ovisi o aktivnosti "Definiranje skale ocjena", koja definira raspone za postizanje određene ocjene u postocima. U slučaju da skala nije definirana, koristi se unaprijed zadana skala.

#### **4.4.1.8. Ostali prošireni i uključeni slučajevi uporabe**

Nakon što je ispit ocijenjen, pokreće se aktivnost "Prikaz rezultata", te po potrebi aktivnosti "Prikaz greške" i "Spremanje rezultata na disk". Zadnje dvije aktivnosti su opcionalne. Spremanje na disk korisnik može uključiti ili isključiti, dok se greške prikazuju samo ako postoje razlike između detektiranih odgovora za ocjenjivanje i točnih odgovora. To mogu biti razlike poput različitog broja pitanja i odgovora i sl.

#### **4.4.1.9. Tijek aktivnosti**

Ovo je pravilan tijek aktivnosti slučaja korištenja "Ocenjivanje ispita":

- 1. Generiranje PDF matrice odgovora**

Ocenjivač definira broj pitanja i odgovora te generira matricu u propisanom formatu.

- 2. Definiranje skale ocjena**

Ocenjivač može definirati vlastitu skalu ocjena. Ako se preskoči, koristi se zadana skala.

- 3. Unos slike matrice točnih odgovora**

Učitava se slika ispravno riješenog ispita kako bi sustav prepoznao točne odgovore.

- 4. Unos slika matrica odgovora za ocjenjivanje**

Učitavaju se slike ispita koje je potrebno ocijeniti.

- 5. Detekcija odgovora**

Sustav obrađuje slike i detektira i klasificira odgovore za svako pitanje.

- 6. Usporedba odgovora**

Detektirani odgovori uspoređuju se s točnim odgovorima na temelju zadane skale ocjena.

- 7. Prikaz rezultata**

Korisniku se prikazuju ocjene i vizualna oznaka točnih/netočnih odgovora.

- 8. Spremanje rezultata na disk (ako je uključena opcija)**

Rezultati se mogu trajno pohraniti, ako je uključena ta opcija.

- 9. Prikaz greške (samo ako je potrebno)**

Ako postoji nesukladnost između matrica, poput razlike u broju pitanja ili odgovora, korisnik će biti obavješten o nastanku greške.

## 5. Dizajn i izrada artefakta

Kroz ostatak rada fokus će biti na dizajnu i izradi artefakta za opisani problem. Artefakt će biti prikazan kroz različite dijagrame i zanimljive isječke koda.

### 5.1. Odabранe tehnologije

Prije prikaza samog rješenja, važno je razumjeti koje su tehnologije odabrane i koji su razlozi za njihov odabir. U prijašnjim poglavljima spomenute su relevantne tehnologije, odnosno biblioteke za računalni vid. Analizirane biblioteke mogu se podijeliti na biblioteke za obradu slike i biblioteke za strojno i duboko učenje. Kroz ovaj rad fokus će biti na izradi rješenja isključivo korištenjem obrade slike. Ta odluka donesena je iz osobnih preferencija. Time se izbor automatski sužava na dvije popularne biblioteke: OpenCV i Scikit-Image.

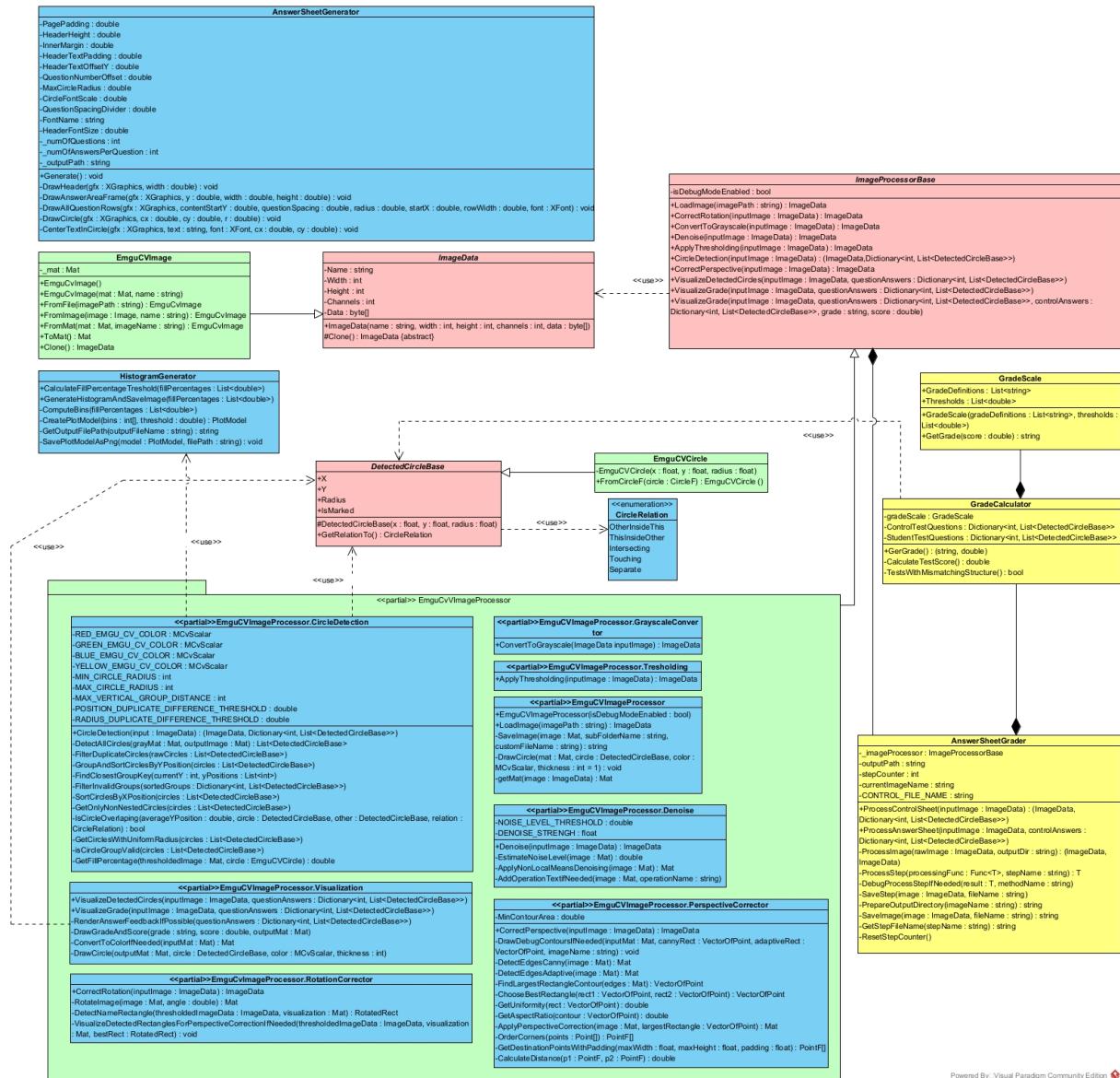
Osim biblioteke, važan je i programski jezik koji je odabran. Budući da je cilj ovog rada izraditi biblioteku otvorenog koda, donesena je odluka koristiti C# i .NET. Taj programski jezik omogućuje izradu biblioteke u obliku .dll datoteke, kao i jednostavnu izradu demo aplikacije korištenjem WinForms tehnologije, koja je također bazirana na C#-u i .NET-u.

Odabir programskog jezika ujedno je olakšao i odabir biblioteke za računalni vid. Budući da Scikit-Image ne podržava odabранe tehnologije, odlučeno je koristiti EmguCV. EmguCV je zapravo .NET omotač za OpenCV biblioteku. Odabir EmguCV-a omogućio je korištenje sve postojeće dokumentacije, materijala i više od 2500 metoda vezanih za OpenCV.

Važno je napomenuti da je prije početka dizajna artefakta donesena odluka da sustav mora biti neovisan o konkretnoj biblioteci za računalni vid. To znači da će se korisnicima omogućiti izrada vlastitih implementacija koristeći bilo koju drugu biblioteku za računalni vid.

Kako bi to bilo moguće, odlučeno je koristiti uzorak dizajna injektiranja ovisnosti (eng. dependency injection). Točnije, definirane su apstraktne klase koje korisnici mogu naslijediti i implementirati prema vlastitim potrebama, a zatim ih injektirati u glavnu klasu biblioteke ovog rješenja.

## 5.2. Strukturni model sustava



Slika 18: Prikaz dijagrama klasa (Vlastita izrada)

Na priloženoj slici prikazana je struktura izrađenog sustava za detekciju odgovora na pismenim ispitima nazvanog GradeVision. Struktura je prikazana korištenjem dijagrama klasa, što omogućuje detaljan uvid u atribute, metode i međuvisnosti svake klase. Klase na ovoj slici možemo podijeliti na klase koje se bave procesiranjem slika i klase temeljne poslovne logike. Nadalje, klase za procesiranje slika mogu se podijeliti na apstraktne i specifične klase.

Apstraktne klase (označene crvenom bojom) su: `ImageData`, `DetectedCircleBase` i `ImageProcessorBase`. U ovom slučaju, te klase postoje kako bi ispunile jedan od nefunkcionalnih zahtjeva (NFZ-02). Taj je zahtjev bio vezan uz jednostavnu zamjenu biblioteka za računalni vid. Specifične klase (označene zelenom bojom) su: `EmguCvImage`, `EmguCvCircle` i `EmguCvImageProcessor`. One predstavljaju zadani implementaciju apstraktnih klasa, napi-

sanu korištenjem biblioteke EmguCV za računalni vid.

Važno je napomenuti da klasa `EmguCvImageProcessor` nasljeđuje apstraktnu klasu `ImageProcessorBase`, ali ne implementira sve njezine metode u jednoj C# datoteci. Metode su implementirane u više datoteka koristeći koncept parcijalnih klasa u programskom jeziku C#. To smanjuje potrebu za implementacijom više sučelja prilikom kreiranja nove implementacije. Klasa `EmguCvImageProcessor` definirana je kroz sljedeće datoteke, pri čemu svaka datoteka poštuje načelo jedinstvene odgovornosti (*eng. Single Responsibility Principle*):

- **EmguCVImageProcessor.cs**
  - Klasa koja sadrži konstruktor te sve zajedničke i pomoćne metode koje nisu specifične samo za jedan korak obrade
- **EmguCVImageProcessor.CircleDetection.cs**
  - Sadrži logiku za detekciju krugova, njihovo grupiranje, filtriranje i klasificiranje
- **EmguCVImageProcessor.Denoise.cs**
  - Sadrži logiku za uklanjanje šuma u slučaju da je razina šuma iznad definiranog praga
- **EmguCVImageProcessor.GrayscaleConvertor.cs**
  - Sadrži logiku za pretvorbu slike u boji u sivu sliku
- **EmguCVImageProcessor.PerspectiveCorrector.cs**
  - Sadrži logiku za detekciju papira ili unutarnjih okvira ispita te ispravljanje perspektivne deformacije
- **EmguCvImageProcessor.RotationCorrector.cs**
  - Sadrži logiku za određivanje orijentacije papira i korekciju pogrešne rotacije
- **EmguCVImageProcessor.Tresholding.cs**
  - Sadrži logiku za pretvorbu sive slike u binarnu sliku korištenjem tehnike adaptivnog praga (*engl. Adaptive Thresholding*)
- **EmguCVImageProcessor.Visualization.cs**
  - Sadrži logiku za vizualizaciju označenih odgovora i ocjena

Svi koraci procesiranja slike definirani su unutar klase `ImageProcessorBase`, a implementirani u klasi `EmguCvImageProcessor`. Važno je naglasiti da u tim klasama nije definiran ni redoslijed operacija ni logika ocjenjivanja testova. Taj je dio izdvojen u klase (označene žutom bojom): `AnswerSheetGrader`, `GradeCalculator` i `GradeScale`. To su ujedno i klase s kojima bi korisnici biblioteke trebali imati najviše interakcije.

### 5.2.1. Klase `ImageData` i `EmguCvImage`

Kako bi se omogućila zamjena biblioteke za računalni vid, bilo je potrebno koristiti podatkovni model slike koji nije vezan ni uz jednu konkretnu biblioteku. Zbog toga je kreirana apstraktna klasa `ImageData`, koja sadrži sve potrebne podatke za daljnje procesiranje slike. To uključuje naziv datoteke, širinu, visinu, broj kanala, podatke o slici u obliku niza bajtova te metodu za kloniranje. Na taj način svaka implementacija tog sučelja može biti predstavljena na isti način.

Na primjer, `EmguCvImage` je implementacija koja predstavlja objekt tipa `Mat` iz biblioteke EmguCV u obliku `ImageData`. Također sadrži razne pomoćne metode poput `ToMat`, `FromMat`, `FromFile` i sličnih, što omogućuje još jednostavniji razvoj.

`ImageData` se koristi u svim dijelovima koda koji moraju biti neovisni o biblioteci za računalni vid, dok se `EmguCvImage` koristi isključivo u specifičnoj implementaciji koja koristi biblioteku EmguCV.

### 5.2.2. Klase `DetectedCircleBase` i `EmguCvCircle`

`DetectedCircleBase` je klasa koja je također nastala iz potrebe da sustav bude neovisan o specifičnoj biblioteci za računalni vid. Njezina je svrha pohranjivanje svih podataka o detektiranim krugovima, dok se specifične implementacije poput `EmguCvCircle` brinu za prikaz specifičnih objekata biblioteke računalnog vida u tom obliku.

`DetectedCircleBase` sadrži sve važne informacije o detektiranom krugu poput: x koordinate, y koordinate, radiusa i vrijednosti koja označava je li krug ispunjen. Osim toga, sadrži i metodu koja provjerava odnos jednog objekta tipa `DetectedCircleBase` u odnosu na drugi objekt iste klase. Time se može saznati radi li se o krugu koji se preklapa i slično, što će u dalnjem dijelu rada biti iznimno važno.

`EmguCvCircle` samo nasljeđuje `DetectedCircleBase` i omogućuje kreiranje objekta na temelju modela podataka `CircleF`, koji je prisutan u biblioteci EmguCV.

### 5.2.3. Klasa `EmguCvImageProcessor`

Kao što je već spomenuto, `EmguCvImageProcessor` je konkretna klasa koja implementira metode naslijedene iz apstraktne klase `ImageProcessorBase`. Budući da se radi o parcijalnoj klasi, metode su implementirane u različitim datotekama.

Može se reći da je ova klasa zapravo jezgra ovog rješenja. Ona je odgovorna za sve korake koji spadaju pod preprocesiranje, procesiranje i dio postprocesiranja.

Najvažniji dijelovi logike provode se u fazi preprocesiranja, budući da se ondje nalazi sva logika koja priprema sliku za daljnju obradu. U toj fazi također su implementirani mehanizmi koji omogućuju da sustav ispravno radi i u uvjetima koji nisu idealni.

Kasnije u radu prikazat će se točan redoslijed tih operacija, koji je definiran klasom

`AnswerSheetGrader`. Također će biti prikazani zanimljivi i nestandardni dijelovi implementacije za svaki od tih koraka.

Sam redoslijed operacija namjerno nije definiran unutar klase `EmguCvImageProcessor`, budući da je ideja bila izdvojiti što više poslovne logike izvan klase koje ovise o određenoj biblioteci za računalni vid. Time je omogućeno da se u budućnosti, po potrebi, napiše nova konkretna implementacija klase `ImageProcessorBase` bez potrebe za ponovnim pisanjem cijele poslovne logike. Može se reći da je u ovom slučaju `EmguCvImageProcessor` "motor" GradeVision sustava, dok je `AnswerSheetGrader` "vozač".

#### 5.2.4. Klase GradeCalculator i GradeScale

Klase `EmguCvImageProcessor` na kraju obrade slike vraća listu odgovora po pitanjima. Ta lista se zatim predaje klasi `GradeCalculator`, koja je zadužena za izračun bodova i postotka točnosti ispita, a potom uz pomoć klase `GradeScale` određuje ocjenu koja će se ispitu dodijeliti.

`GradeScale` predstavlja unaprijed definiranu skalu ocjena, ali postoji i mogućnost definiranja vlastite skale, što ocjenjivačima pruža dodatnu fleksibilnost.

Unutar ove dvije klase enkapsulirana je sva logika za ocjenjivanje ispita. Trenutna implementacija podržava ocjenjivanje pitanja s jednostrukim i višestrukim odgovorima. Proses bodovanja temelji se na strogoj logici: bod se dodjeljuje samo ako je odgovor u potpunosti točan, to jest nema parcijalnih bodova ni penalizacije za djelomične odgovore.

Važno je napomenuti da ove dvije klase također sudjeluju u fazi postprocesiranja, točnije u konačnom koraku Obrada izvučenih podataka, gdje koriste prethodno definiranu bazu znanja (listu točnih odgovora iz kontrolnog ispita). Nakon toga se ocjene vizualiziraju, a za tu funkcionalnost ponovno je zadužena klasa `EmguCvImageProcessor`.

#### 5.2.5. Ostale klase

Od ostalih klasa postoje `HistogramGenerator` i `AnswerSheetGenerator` klase. Radi se o generatorima sadržaja. Klasa `AnswerSheetGenerator` odgovorna je za generiranje matrice odgovora prema zadanim parametrima, kao što je definirano u funkcionalnom zahtjevu FZ-02. Za razliku od ostalih klasa, riječ je o potpuno neovisnoj komponenti sustava, čija se metoda `Generate()` može pozvati u bilo kojem trenutku izvođenja programa, bez utjecaja na njegovo daljnje izvođenje.

S druge strane, klasa `HistogramGenerator` je pomoćna klasa koja ima ključnu ulogu u procesu detekcije krugova. Implementira algoritam za određivanje praga koji definira je li neki odgovor označen ili nije. Izračunati pragovi se zatim vizualiziraju i pohranjuju u obliku histograma na disk. Histogrami se spremaju samo kada je program pokrenut u načinu rada za otklanjanje pogrešaka (*engl. debug mode*), koji je definiran u funkcionalnom zahtjevu FZ-06.

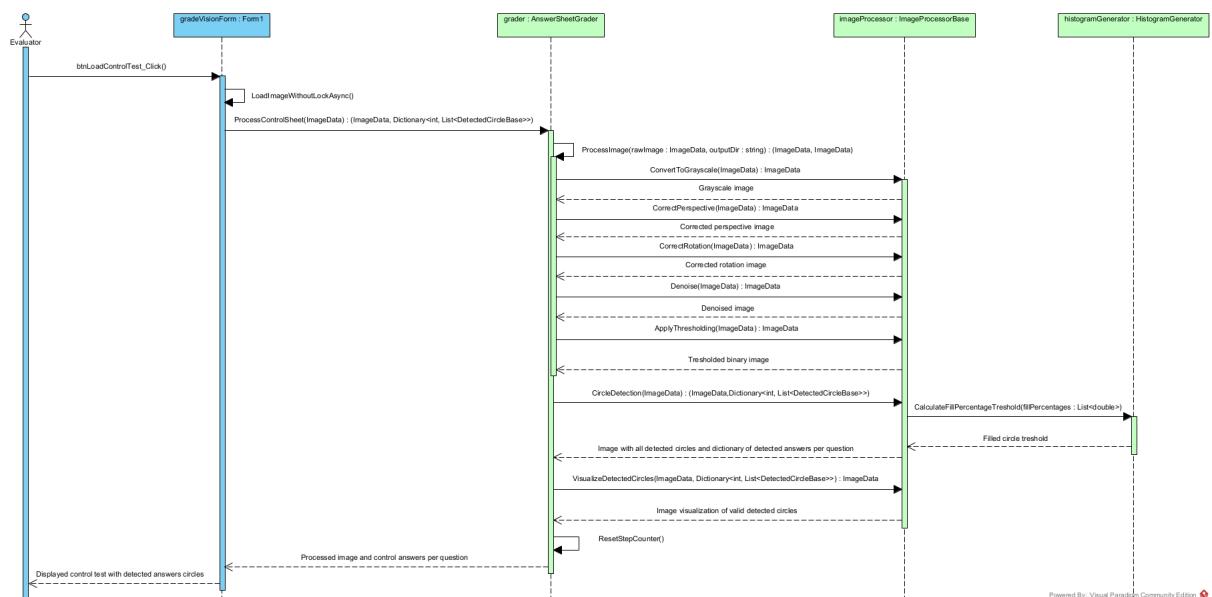
Važno je napomenuti da je klasa `HistogramGenerator` potpuno neovisna o bibliotekama za računalni vid, iako se primarno koristi od strane `EmguCVImageProcessor` klase.

Ona prima listu decimalnih brojeva koji predstavljaju odgovore na ispitnim listovima s određenim postotkom ispunjenosti. Te podatke obrađuje i vraća prag ispunjenosti kruga u obliku postotka. Ako je postotak ispunjenosti viši od izračunatog praga, krug se smatra označenim.

## 5.3. Modeliranje ponašanja sustava

Kroz sljedeće dijagrame slijeda prikazat će se interakcije između korisnika i različitih dijelova sustava na visokoj razini, dok će se kroz daljnji rad prikazati zanimljivi dijelovi detaljnije. Svaki dijagram slijeda vezan je uz jedan od koraka koje ocjenjivač može napraviti i time pokrenuti neku interakciju sa sustavom.

### 5.3.1. Ponašanje sustava tijekom detekcije odgovora s kontrolnog ispita



Slika 19: Prikaz dijagrama slijeda za detekciju odgovora s kontrolnog ispita (Vlastita izrada)

Dosad se GradeVision spominjao samo kao biblioteka, ali za potrebe testiranja i demonstracije izrađena je jednostavna WinForms aplikacija koja koristi mogućnosti te biblioteke, a koja će se u dalnjem dijelu rada spominjati kao GradeVision korisničko sučelje, dok će se biblioteka spominjati kao GradeVisionLib. Također, u dalnjem dijelu rada spominjat će se pojmovi kontrolni ispit i ispit za ocjenjivanje. Kontrolni ispit označava ispit koji sadrži sve točne odgovore za svako pitanje, dok ispit za ocjenjivanje označava ispit čija je točnost nepoznata te se tek treba utvrditi usporedbom detektiranih odgovora s detektiranim odgovorima kontrolnog ispita (eng. control test).

Priloženi dijagram prikazuje proces detekcije odgovora s kontrolnog ispita. To je vrlo važan proces, budući da je njegov konačni rezultat izrada baze znanja s točnim odgovorima za svako pitanje, koja se kasnije koristi pri ocjenjivanju.

Taj proces pokreće ocjenjivač tako da učita sliku kontrolnog ispita u GradeVision korisničko sučelje. Zatim aplikacija GradeVision poziva AnswerSheetGrader, koji je dio GradeVisionLib-a. Točnije, poziva se metoda ProcessControlSheet (ImageData), čime se zapravo pokreće obrada slike. Veći dio obrade slike izvodi se u metodi ProcessImage (ImageData rawImage, string outputDir), koja orkestrira sve aktivnosti preprocesiranja pozivanjem metoda konkretnе implementacije klase ImageProcessorBase sljedećim redoslijedom:

- ConvertToGrayscale(image)
- CorrectPerspective(image)
- CorrectRotation(image)
- Denoise(image)
- ApplyThresholding(image)

Naravno ove metode kasnije pozivaju svoje privatne pomoćne metode no to će uvjek ovisiti o biblioteci računalnog vida koja se koristi pa taj dio nije prikazan na dijagramu. Metoda ProcessImage (ImageData rawImage, string outputDir) se također koristi i kod preprocesiranja ispita za ocijenjivanje što će u nastavku biti prikazano. Važno je napomenuti da je sam redoslijed aktivnosti preprocesiranja dobiven kroz opsežno testiranje i iteriranje tijekom razvoja i ova varijanta se pokazala najboljom. Kvaliteta redoslijeda tih aktivnosti bila je mjerena prema preciznosti samog sustava nad skupom podataka vlastite izrade koji će također biti priloženi uz ovaj rad.

Osim toga, metoda ProcessImage (ImageData rawImage, string outputDir) se također koristi pri preprocesiraju ispita za ocijenjivanje, što će biti prikazano u nastavku. Naravno, ove metode kasnije pozivaju svoje privatne pomoćne metode, što će uvjek ovisiti o biblioteci računalnog vida koja se koristi, pa taj dio nije prikazan na dijagramu zbog čitljivosti.

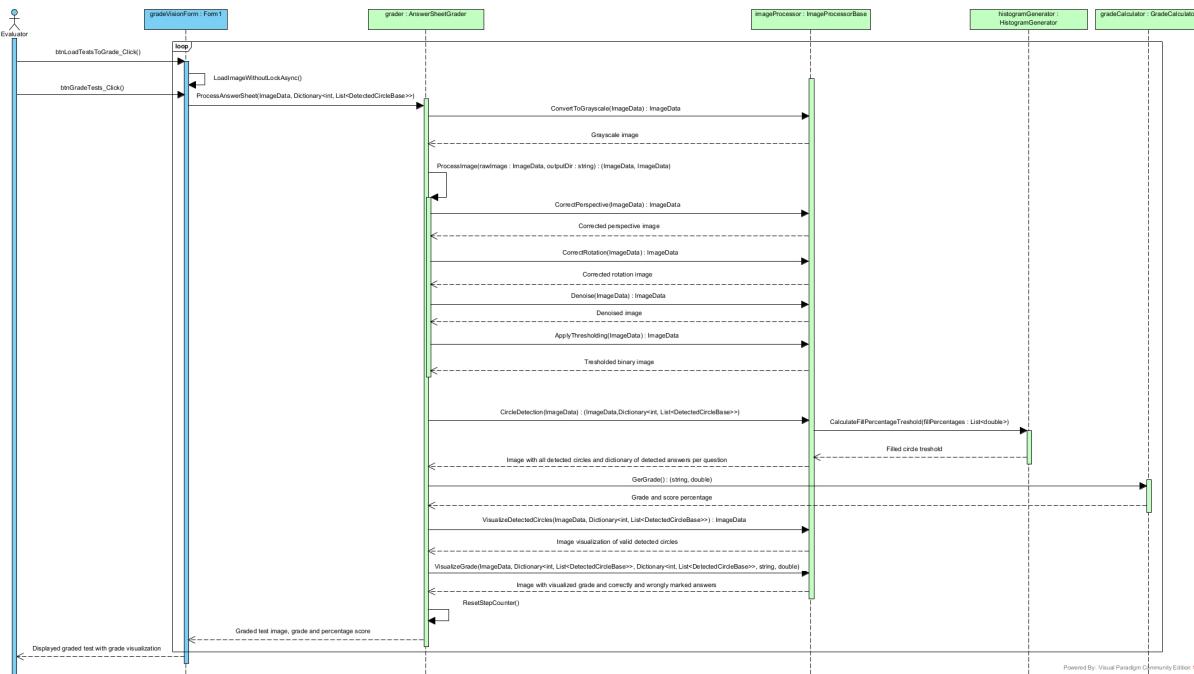
Nakon što je preprocesiranje završeno, započinje faza procesiranja tako što AnswerSheetGrader poziva metodu CircleDetection (ImageData inputImage). Metoda CircleDetection jedan je od najvažnijih dijelova ovog rada, budući da se u njoj odvija detekcija odgovora i njihovo klasificiranje u dvije klase: označen i neoznačen odgovor.

Sama logika klasificiranja koristi opisne vrijednosti ispunjenosti objekta, koje izražavaju postotak u kojem je pojedini odgovor ispunjen. Međutim, samo poznavanje postotka ispunjenosti nije dovoljno za ispravnu klasifikaciju odgovora. Potrebno je odrediti prag ispunjenosti koji razdvaja te dvije klase. Taj se prag može postaviti proizvoljno, no takav pristup nije pouzdan jer ne ispunjava svaka osoba krug odgovora na isti način. Osim toga, dodatni čimbenici poput veličine kruga odgovora na papiru također utječu na rezultate.

Zbog toga je bilo potrebno osmisliti preciznije rješenje. Ono je implementirano u klasi HistogramGenerator, unutar metode CalculateFillPercentageThreshold (List<double> fillPercentages). Upravo se ta metoda poziva unutar CircleDetection metode prije nego što se vrati slika sa svim detektiranim odgovorima i rječnik klasificiranih odgovora grupiranih po pitanjima.

Klase `AnswerSheetGrader` zatim poziva metodu `VisualizeDetectedCircles` iz klase `ImageProcessorBase` kako bi se klasificirani odgovori vizualizirali. Nakon toga se kreirana slika vraća sve do GradeVision korisničkog sučelja, gdje je prikazana ocjenjivaču.

### 5.3.2. Ponašanje sustava tijekom ocjenjivanja



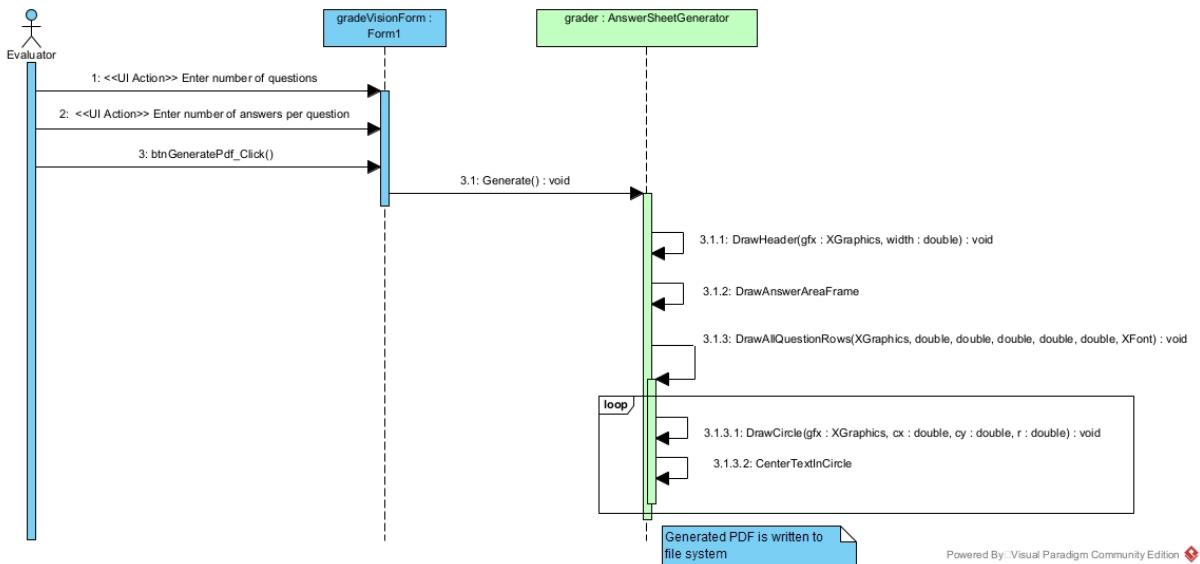
Slika 20: Prikaz dijagrama slijeda ocjenjivanja ispita (Vlastita izrada)

Priloženi dijagram ima mnogo sličnosti sa prijašnjim dijagramom slijeda, ali razlikuje se u par ključnih stvari. Prva razlika je da korsnik ne unosi samo jednu sliku u sustav već više njih. Zatim se poziva metoda `ProcessAnswerSheet (ImageData inputImage, Dictionary<int, List<DetectedCircleBase>> controlAnswers, GradeScale gradeScale)` koja procesira ispite za ocjenjivanje. Preprocesiranje ponovno provodi `ProcessImage (ImageData rawImage, string outputDir)` metoda već spomenutim koracima i redoslijedom. Zatim se provodi detektiranje krugova koje je također već spomenuto, no tu sve sličnosti sa prijašnjim dijagramom završavaju. Nakon detektiranja krugova poziva se `GradeCalculator` klasa, točnije njezina metoda `GetGrade ()`.

Nakon što metoda vrati ocjenu i postotak rješenosti, poziva se sljedeći korak. Taj korak uključuje vizualizaciju ocjene na način da se na ispitu označe točno i krivo označeni odgovori te da se u gornjem desnom kutu slike napiše postotak i ocjena. Zatim se slika sa vizualizacijom ocjene vraća na ekran ocjenjivača i ponavlja se isti proces za ostale ispite koji još nisu ocijenjeni.

Važno je napomenuti da se ovaj slijed ne može pokrenuti dok god se ne izvrše svi koraci iz prvog dijagrama slijeda za kontrolni ispit. Također, generalna struktura ispita za ocjenjivanje se mora poklapati sa kontrolnim ispitom, pod to spada isti broj pitanja i odgovora.

### 5.3.3. Dijagram slijeda – Generiranje matrice odgovora



Slika 21: Prikaz dijagrama slijeda za generiranje PDF matrice odgovora (vlastita izrada)

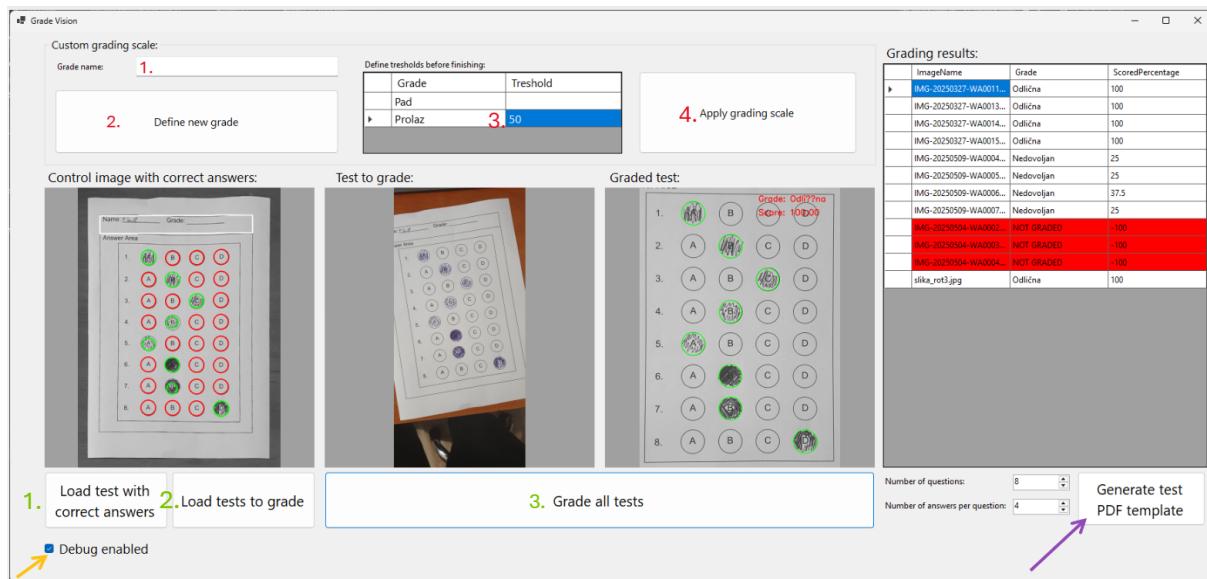
Kroz priloženi dijagram slijeda može se vidjeti da je samo generiranje matrice odgovora u obliku PDF-a prilično jednostavno. Kako bi se PDF generirao, korisnik najprije mora definirati ukupan broj pitanja i broj ponuđenih odgovora po pitanju. Nakon što su ti parametri definirani, pritišće se gumb za generiranje PDF-a, čime se daljnja kontrola nad izvođenjem predaje klasi forme u WinForm aplikaciji. Ta klasa zatim poziva metodu `Generate()` klase `AnswerSheetGenerator`.

Metoda `Generate` je metoda bez povratnog tipa podataka, što nije problem jer ona izravno generirani PDF spremi na disk. Metoda `Generate` poziva svoje privatne metode sljedećim redoslijedom:

- `DrawHeader` – Crta pravokutni okvir na vrhu stranice i ispisuje tekst s poljima za ime i ocjenu osobe koja piše ispit.
- `DrawAnswerAreaFrame` – Crta pravokutni okvir koji označava područje u kojem se nalaze pitanja i odgovori.
- `DrawAllQuestionRows` – Ispisuje redove s brojevima pitanja i pripadajućim krugovima za svaki ponuđeni odgovor.
  - `DrawCircle` – Crta jedan krug koji predstavlja ponuđeni odgovor.
  - `CenterTextIntoCircle` – Centrira i ispisuje pripadajuće slovo unutar prethodno nacrtanog kruga.

## 5.4. GradeVision aplikacija

Cilj ovog rada je izrada biblioteke za detekciju odgovora, no kako bi se ista mogla testirati i demonstrirati, izrađena je GradeVision stolna aplikacija. Aplikacija je izradena korištenjem .NET Core, C# i WinForms tehnologija. Te su tehnologije odabране zbog jednostavnosti i brzine izrade demo aplikacije, ali i zato što je i GradeVision biblioteka napisana korištenjem .NET Core i C#.



Slika 22: Prikaz korisničkog sučelja GradeVision stolne demo aplikacije (vlastita izrada)

Korisničko sučelje s priložene slike možda na prvi pogled djeluje komplikirano, no zapravo je prilično jednostavno i sadrži sve potrebne elemente za testiranje i demonstraciju rada biblioteke. Glavni elementi korisničkog sučelja su tri okvira za prikaz slika. Prvi okvir prikazuje kontrolni ispit koji se, nakon učitavanja i ocjenjivanja, prikazuje na tom mjestu. U drugom okviru nalazi se izvorna slika koja se ocjenjuje, dok se desno od nje prikazuje ocijenjena verzija te slike. Također, s desne strane nalazi se i tablica koja prikazuje povijest ocjenjivanja.

Osim tih elemenata, u gornjem lijevom kutu nalaze se kontrole za definiranje nove skale ocjenjivanja, ako je to potrebno. U donjem desnom kutu nalaze se parametri matrice odgovora i gumb za njezino generiranje. Također u dolnjem lijevom kutu postoji gumb za uključivanje i isključivanje načina rada za otklanjanje grešaka.

### 5.4.1. Način rada

## 5.5. GradeVision biblioteka

prolazak kroz kod

## **6. Evaluacija**

### **6.1. Skup testnih podataka**

#### **6.1.1. Prikaz koraka obrada slike nad testnim podatcima**

### **6.2. Manualno testiranje rješenja**

## **7. Diskusija**

### **7.1. Analiza rezultata**

### **7.2. Limitacije rješenja**

### **7.3. Poboljšanja rješenja**

## **8. Zaključak**

# Popis literature

- [1] A. Bloomfield, „Evolution of a digital paper exam grading system,” *2010 IEEE Frontiers in Education Conference (FIE)*, IEEE, str. 27–30. DOI: 10.1109/FIE.2010.5673292.
- [2] sabbott, „Standardized Test Definition,” *Glossary of Education Reform*, studeni 2015. adresa: <https://www.edglossary.org/standardized-test>.
- [3] S. Leonard, „Automated Grading for Subjective Assessments: Challenges and Solutions,” TAO, veljača 2025. adresa: <https://www.taotesting.com/blog/automated-grading-for-subjective-assessments-challenges-and-solutions>.
- [4] E. Miguel de Elias, P. Tasinazzo i R. Hirata Jr, „Optical Mark Recognition: Advances, Difficulties, and Limitations,” *SN Computer Science*, sv. 2, rujan 2021. DOI: 10.1007/s42979-021-00760-z.
- [5] M. E. Alam, F. Akter, T. Alam, S. Farid, M. I. Haque i M. Arefin, „Approaching Computer Vision And Image Processing Technique For Optical Mark Recognition Sheet Scan And Verification,” *2022 International Conference on Recent Progresses in Science, Engineering and Technology (ICRPSET)*, 2022., str. 1–5. DOI: 10.1109/ICRPSET57982.2022.10188569.
- [6] G. Smith, R. Haworth i S. Zitnik, „Computer Science Meets Education: Natural Language Processing for Automatic Grading of Open-Ended Questions in eBooks,” *Journal of Educational Computing Research*, sv. 58, svibanj 2020. DOI: 10.1177/0735633120927486.
- [7] Saiwa, *Top 10 Computer Vision Libraries |A Comprehensive and Detailed Guide*. China: Medium, studeni 2024., ISBN: 978-745304301. adresa: <https://medium.com/@saiwadotai/top-10-computer-vision-libraries-a-comprehensive-and-detailed-guide-745c304e3016>.
- [8] *What Is a Digital Image? - Ricoh Scanners*, [Online; pristupljeno 13. travnja 2025.], travanj 2025. adresa: <https://www.pfu-us.ricoh.com/blog/what-is-a-digital-image>.
- [9] *Understanding Aspect Ratios in Photography | Adobe Express*, [Online; pristupljeno; 30. travnja 2025.], travanj 2025. adresa: <https://www.adobe.com/express/discover/sizes/photo-aspect-ratio>.
- [10] *Color photography history, tips, and techniques - Adobe*, [Online; accessed 30. Apr. 2025], travanj 2025. adresa: <https://www.adobe.com/creativecloud/photography/type/color-photography.html>.

- [11] *Grayscale Image - an overview* | *ScienceDirect Topics*, [Online; pristupljeno 20. travnja 2025.], travanj 2025. DOI: 10.1016/B978-0-12-822271-3.00013-X.
- [12] linkgit, „Why use Grayscale Conversion when Processing Images?” *Grayscale Images*, svibanj 2024. adresa: <https://www.grayscaleimage.com/why-use-grayscale-conversion-when-processing-images>.
- [13] *Grayscale Image - an overview* | *ScienceDirect Topics*, [Online; pristupljeno 16. travnja 2025.], travanj 2025. DOI: 10.1016/B978-0-12-822271-3.00013-X.
- [14] *why we should use gray scale for image processing*, [Online; pristupljeno 20. travnja 2025.], travanj 2025. adresa: <https://stackoverflow.com/questions/12752168/why-we-should-use-gray-scale-for-image-processing>.
- [15] *Image Processing: Techniques, Types, & Applications [2024]*, [Online; pristupljeno 21. travnja 2025.], travanj 2025. adresa: <https://www.v7labs.com/blog/image-processing-guide>.
- [16] [Online; pristupljeno 21. travnja 2025.], travanj 2025. adresa: <https://dl.icdst.org/pdfs/files4/01c56e081202b62bd7d3b4f8545775fb.pdf>.
- [17] M. Patel, „The Complete Guide to Image Preprocessing Techniques in Python,” *Medium*, listopad 2023., ISSN: 3080-4550. adresa: <https://medium.com/@maahip1304/the-complete-guide-to-image-preprocessing-techniques-in-python-dca30804550c>.
- [18] *Image Acquisition* | *Cloudinary*, [Online; pristupljeno 22. travnja 2025.], travanj 2025. adresa: <https://cloudinary.com/glossary/image-acquisition>.
- [19] *What is noise in photography*, [Online; pristupljeno 22. travnja 2025.], travanj 2025. adresa: <https://www.adobe.com/creativecloud/photography/hub/guides/what-is-noise-in-photography.html>.
- [20] A. Swain, „Noise filtering in Digital Image Processing - Anisha Swain - Medium,” *Medium*, srpanj 2023. adresa: <https://medium.com/@anishaswain/noise-filtering-in-digital-image-processing-d12b5266847c>.
- [21] *Caption for Median Filtering Illustration*, [Online; pristupljeno 22. travnja 2025.], listopad 2022. adresa: <https://visns.neocities.org/MedianFiltering>.
- [22] *Image preprocessing and enhancement* | *Geospatial Engineering Class Notes* | *Fiveable*, [Online; pristupljeno 24. travnja 2025.], travanj 2025. adresa: <https://fiveable.me/geospatial-engineering/unit-4/image-preprocessing-enhancement/study-guide/L5fa2eI0Xk8chrKd>.
- [23] R. Shaikh, „OpenCv Perspective Transformation - Analytics Vidhya - Medium,” *Medium*, veljača 2024. adresa: <https://medium.com/analytics-vidhya/opencv-perspective-transformation-9edffefb2143>.
- [24] R. Pulapakura, „Image Segmentation — A Beginner’s Guide | Medium,” *Medium*, veljača 2024. adresa: <https://medium.com/@raj.pulapakura/image-segmentation-a-beginners-guide-0ede91052db7>.

- [25] P. Chhikara, „Understanding Morphological Image Processing and Its Operations | Towards Data Science,” *Towards Data Science*, siječanj 2025. adresa: <https://towardsdatascience.com/understanding-morphological-image-processing-and-its-operations-7bcf1ed11756>.
- [26] [Online; pristupljeno 25. travnja 2025.], studeni 2022. adresa: [https://cmrcet.ac.in/files/ECE/ececoursefile/13.pdf?utm\\_source=chatgpt.com](https://cmrcet.ac.in/files/ECE/ececoursefile/13.pdf?utm_source=chatgpt.com).
- [27] M. Abramov, „Semantic Segmentation vs Object Detection: Differences | Keymakr,” *Keymakr*, veljača 2025. adresa: <https://keymakr.com/blog/semantic-segmentation-vs-object-detection-understanding-the-differences>.
- [28] P. Sharma, „Image Classification vs. Object Detection vs. Image Segmentation,” *Medium*, prosinac 2021. adresa: <https://medium.com/analytics-vidhya/image-classification-vs-object-detection-vs-image-segmentation-f36db85fe81>.
- [29] siromer, „Contour Detection using OpenCV - siromer - Medium,” *Medium*, siječanj 2025. adresa: <https://medium.com/@siromermer/contour-detection-using-opencv-detecting-and-counting-coins-2d5192597e3c>.
- [30] *Post-processing and visualization | Applications of Scientific Computing Class Notes | Fiveable*, [Online; pristupljeno 28. travnja 2025.], travanj 2025. adresa: <https://library.fiveable.me/applications-of-scientific-computing/unit-8/post-processing-visualization/study-guide/6jrpX4V64bjrgSLQ>.
- [31] R. Sheldon, „image compression,” *WhatIs*, srpanj 2022. adresa: <https://www.techtarget.com/whatis/definition/image-compression>.
- [32] MenyePy, „2 free open-source OMR grading projects on github,” *DEV Community*, veljača 2022. adresa: <https://dev.to/menyepy/2-free-open-source-omr-projects-on-github-2dl3>.
- [33] [v1] *Diagrams · Udayraj123/OMRChecker Wiki*, [Online; pristupljeno 29. travnja 2025.], travanj 2025. adresa: <https://github.com/Udayraj123/OMRChecker/wiki/%5Bv1%5D-Diagrams>.
- [34] *open-mcr*, [Online; accessed 25. May 2025], svibanj 2025. adresa: <https://github.com/iansom5653/open-mcr>.
- [35] S. Mondal, P. De, S. Malakar i R. Sarkar, „OMRNet: A lightweight deep learning model for optical mark recognition,” en, *Multimedia Tools and Applications*, srpanj 2023., ISSN: 1380-7501, 1573-7721. DOI: 10.1007/s11042-023-15408-8. pogledano 13. srpnja 2023. adresa: <https://link.springer.com/10.1007/s11042-023-15408-8>.

# Popis slika

1.	Prikaz slike u boji i pripadajućih kanala boja u RGB modelu (vlastita izrada) . . . . .	8
2.	Prikaz sive slike (vlastita izrada) . . . . .	9
3.	Prikaz binarne slike (vlastita izrada) . . . . .	10
4.	Usporedba svjetlosne komponente i komponenta boja(vlastita izrada) . . . . .	11
5.	Primjer medijanskog filtera [21] . . . . .	14
6.	Usporedba procesirane slike bez i s pretpresiranjem (vlastita izrada) . . . . .	15
7.	Prikaz procesa ispravka perspektive subjekta na slici [23] . . . . .	16
8.	Usporedba tipova segmentacije [24] . . . . .	17
9.	Prikaz erozije kao morfološke operacije [25] . . . . .	18
10.	Prikaz dilatacije kao morfološke operacije [25] . . . . .	18
11.	Prikaz kompozitnih morfoloških operacija [25] . . . . .	19
12.	Usporedba segmentacije i detekcije objekta [28] . . . . .	20
13.	Prikaz rezultata detekcije krugova pomoću kontura [28] . . . . .	21
14.	Prikaz dijela toka procesiranja slike kod OMRChecker-a [33] . . . . .	24
15.	Prikaz pomoćnih oznaka za detektiranje kutova matrice odgovora i orijentacije [33] [34] . . . . .	25
16.	Prikaz strukture OMRNet klasifikatora [35] . . . . .	25
17.	Prikaz Use-case diagrama (Vlastita izrada) . . . . .	31
18.	Prikaz dijagrama klase (Vlastita izrada) . . . . .	35
19.	Prikaz dijagrama slijeda za detekciju odgovora s kontrolnog ispita (Vlastita izrada)	39
20.	Prikaz dijagrama slijeda ocijenjivanja ispita (Vlastita izrada) . . . . .	41
21.	Prikaz dijagrama slijeda za generiranje PDF matrice odgovora (vlastita izrada) . .	42
22.	Prikaz korisničkog sučelja GradeVision stolne demo aplikacije (vlastita izrada) . .	43

# **Popis tablica**

1.	Usporedba biblioteka i okvira računalnog vida [7] . . . . .	5
2.	Funkcionalni zahtjevi . . . . .	29
3.	Nefunkcionalni zahtjevi . . . . .	30