Rekonstrukcija virtualnih pogleda dobivenih pomoću DIBR algoritama

Pogač, Tina

Master's thesis / Diplomski rad

2020

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University North / Sveučilište Sjever**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:952352

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-25**

Repository / Repozitorij:

University North Digital Repository

SVEUČILIŠTE SJEVER
SVEUČILIŠNI CENTAR VARAŽDIN
Studij Multimedije

DIPLOMSKI RAD br. 008/MMD/2020

REKONSTRUKCIJA VIRTUALNIH POGLEDA DOBIVENIH POMOĆU DIBR ALGORITAMA

Tina Pogač

Varaždin, rujan 2020

Studij Multimedije

DIPLOMSKI RAD br. 008/MMD/2020

REKONSTRUKCIJA VIRTUALNIH POGLEDA DOBIVENIH POMOĆU DIBR ALGORITAMA

Student:
Tina Pogač 0898/336D

Mentor:
izv. prof. dr. sc. Emil Dumić

Varaždin, rujan 2020

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR-42000 Varaždin

Prijava diplomskog rada

Definiranje teme diplomskog rada i povjerenstva

ODJEL: Odjel za multimediju

STUDIJ: diplomski sveučilišni studij Multimedija

PRISTUPNIK: Pogač Tina | MATIČNI BROJ: 0898/336D

DATUM: 16.06.2020. | KOLEGIJ: Multimedijska videotehnologija

NASLOV RADA: Rekonstrukcija virtualnih pogleda dobivenih pomoću DIBR algoritama

NASLOV RADA NA ENGL. JEZIKU: View synthesis using DIBR algorithms

MENTOR: Emil Dumić | ZVANJE: doc.dr.sc.

ČLANOVI POVJERENSTVA:
1. doc.dr.sc. Andrija Bernik - predsjednik
2. doc.art. Robert Geček - član
3. doc.dr.sc. Emil Dumić - mentor
4. doc.art.dr.sc. Mario Periša - zamjenski član
5.

Zadatak diplomskog rada

BROJ: 008/MMD/2020

OPIS:

U ovom radu će biti opisani i uspoređeni različiti algoritmi za izračun virtualnih pogleda, dobivenih od jednog ili više postojećih pogleda sastavljenih od teksture i dubine (DIBR, Depth Image Based Rendering). Cjelokupna reprezentacija vizualne scene može se opisati plenoptičkom funkcijom. Međutim, u praksi se koriste jednostavniji modeli reprezentacije scene, poput prikaza scene sa više pogleda i pripadajućih dubina. Budući se do krajnjeg gledatelja može prenijeti ograničen broj snimljenih pogleda, bitan je razvoj algoritama za izračun novih, virtualnih pogleda. Novije metode za rekonstrukciju virtualnih pogleda koriste dubinsko učenje pomoću kojeg se i iz samo jednog pogleda (teksture) mogu rekonstruirati virtualni pogledi.
Svaki od snimljenih pogleda se obično pohranjuje u obliku teksture i dubine. Sam postupak za kompresiju takvih videozapisa je podržan u 3D-HEVC standardu, ekstenziji H.265/HEVC (MPEG-H Part 2). Također, budući standard za kompresiju videosadržaja VVC isto podržava kodiranje više pogleda (MPEG-I Part 3, Part 12). Sa korisnikove strane, virtualne poglede je moguće prikazati na 2D monitoru ili 3D monitoru. U radu će biti dan i prikaz različitih načina snimanja, kompresije, prijenosa i prikaza videozapisa s više pogleda.
Praktični dio rada vezan je za ispitivanje različitih načina poboljšanja slika i videozapisa, koji su dobiveni kao virtualni pogledi od jednog ili više postojećih pogleda sastavljenih od teksture i dubine (DIBR). Bit će uspoređeni različiti algoritmi za ispravljanje pogrešaka u slici koje nastaju prilikom izračuna virtualnih pogleda.

ZADATAK URUČEN: 08.6.2020 | POTPIS MENTORA: Emil Dumić

SVEUČILIŠTE SJEVER

Sažetak

Ovim radom će se opisati nekoliko DIBR algoritama te prikazati, na nekoliko slika, kako neki od tih algoritama rade.

Postoje brojne metode kojima se popravljaju izobličenja i greške na slikama ili video zapisima, ovdje je prikazano četiri metode. Rad je popraćen praktičnim dijelom koji kroz program i tri algoritma pokazuje kako se stvaraju novi pogledi i popravljaju greške nastale promjenom kuta gledanja.

Rad također sadržava objektivno istraživanje koje je usporedilo rješenja dobivena tim algoritmima.

Ključne riječi: DIBR, algoritam, popravljanje slika (*eng. inpainting*), 3D, filter, konverzija, BRISQUE, NIQE, PIQE

Summary

This paper describes a couple of DIBR algorithms and, using a few images, it shows how those algirithms work.

There are many methods that are being used to restore warped and damaged images or videos, four of which will be presented here. Apart from describing those methods, this paper will also present a program and three algorithms that show how novel views are created and how damages caused by the change of point of view are being fixed, inpainted.

Key words: DIBR, algorithm, inpainting, 3D, filter, conversion, BRISQUE, NIQE, PIQE

Popis korištenih kratica

AVC	Advanced Video Coding	Napredno Video Kodiranje
ATTEST	Advanced Three-Dimensional Television System Technologies	Napredne trodimenzionalne tehnologije televizijskog sustava
BRISQUE	Blind/Referenceless Image Spatial Quality Evaluator	Slijepi/Bez-referentni evaluator spatijalne kvalitete slike
CI	Contours of Interest	Konture u Interesu
CTBI	Coherence Transport Based Inpainting	Slikanje temeljeno na koherentnom prijevozu
DIBR	Depth Based Image Rendering	Renderiranje slika temeljeno na dubini
FVV	Free Viewpoint Video	Video sa slobodnom točkom gledišta
HVS	Human Visual System	Ljudski vizualni sustav
HEVC	High-Efficiency Video Coding	Video kodiranje s visokom učinkovitosti
MVD	Multi-View Video-plus-depth	Višepogledni video-plus-dubina
NIQE	Naturalness Image Quality Evaluator	Evaluator kvalitete prirodnosti slike
PIQE	Perception Based Quality Evaluator	Evaluator kvalitete temeljen na percepciji
SVD	Scalable Video Coding	Skalabilno Video Kodiranje
VSRS	View Synthesis Reference Software	Referentni softver za sinteziranje pogleda

Sadržaj

1. UVOD 11
2. Stvaranje 3D sadržaja, kompresija i transmisija 12
3. 3D Izobličenje slike 15
4. 2D-u-3D Video Konverzija 17
4.1. Princip konverzije 17
4.2. Monokularni dubinski znakovi u videozapisima 20
4.3. DIBR model za konverziju 2D-u-3D 22
5. Popunjavanje rupa u sintetiziranom pogledu (popunjavanje disokluzija) 24
5.1. Proces 3D izobličenja slike kod sinteze 25
5.2. Disokluzije i problemi 27
5.3. Predprocesiranje dubinskog videa 29
5.3.1. Gaussian Filter 30
5.3.2. Bilateralni Filter 30
5.4. Popunjavanje slika (*eng. Image Inpainting*) 32
5.4.1. Criminsijev algoritam 33
5.4.2. View Synthesis Reference Software (VSRS) 35
6. Praktični rad 36
6.1. Program i kod 36
6.2. Prikaz popravljenih slika pomoću tri algoritma 45
6.2.1. Distance transform + Gaussian Filter 45
6.2.2. Criminisijev algoritam 59
6.2.3. Coherence Transport Based Inpainting 70
6.3. Objektivno (No-Reference) istraživanje 81
6.3.1. Statistička usporedba ocjena 87
6.3.2. Vremenska usporedba popravljanja slike 93
7. Zaključak 94
8. Literatura 96
9. Popis slika 98
10. Popis tablica 101

1. UVOD

DIBR (*Depth Based Image Rendering*) tehnika jedna je od ključnih rješenja i obećavajućih alata za podršku naprednih 3D video usluga, sintetiziranjem nekih novih pogleda iz reprezentacije podataka teksture-plus-dubine ili njenih višepoglednih ekstenzija[1].

Kao osnovni primjer mogu se uzeti dvije kamere (K1 i K2). Obje kamere imaju svoj pogled i zajedničko polje, no K1 ne sadrži neke promatrane točke K2 i obrnuto. Tu nastaju dva scenarija: generiranje drugog pomaknutog pogleda jedne od kamera, sintetiziranje željenih srednjih pogleda od najmanje dva susjedna referentna pogleda kako bi se dobilo slobodno promatranje scene. Kao ranije navedeno, K1 ne sadrži sve promatrane točke koje ima K2 i obrnuto, što je razlog pojave "rupa" u novonastalom pogledu. Te "rupe" nazivaju se disokluzije (*eng. disocclusion*), a njihova veličina ovisi o razmaku između kamera K1 i K2. Veličina disokluzija raste s porastom razmaka između kamera[2].

Nastali problem s disokluzijama prolazi kroz DIBR model (*eng. framework*). DIBR model sastoji se od dvije strategije koje su na različitim mjestima DIBR-ovog dijagrama toka (*eng. flowchart*). Prvo je korištenje niskopropusnog filtra za pre-procesiranje dubine videozapisa prije rekonstrukcije kako bi postigli uklanjanje gore navedenih disokluzija. Sintetizirani pogled je poslije procesiran kako bi popunio veće nedostatke/praznine u području s vjerodostojnim, odgovarajućim informacijama o boji. Proces popunjavanja navedenih disokluzija zove se još popunjavanje rupa (*eng. hole filling*)[2].

Postoji više različitih algoritama kojima se postiže uklanjanje i popunjavanje disokluzija. Odabrani algoritmi detaljno su opisani dalje u radu te uspoređeni.

2. Stvaranje 3D sadržaja, kompresija i transmisija

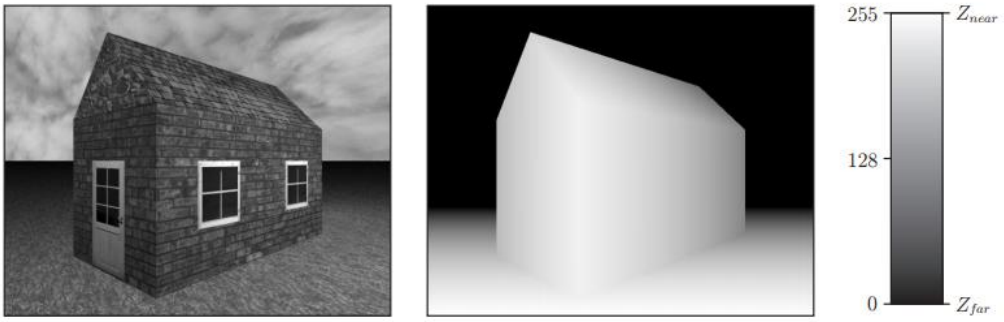
Postoje dva komplementarna pristupa kod stvaranja modernog 3D sadržaja. Jedan pristup koristi simultano snimanje, "hvatanje" videozapisa i videu pripadajuće dubinske informacije. Za takvo snimanje koriste se kamere poput ZcamTM, razvijena od strane 3DV Systems-a, koje u svom snimanju koriste aktivni domet (*eng. active range*) za dobivanje navedenih informacija[3].

Navedeni tip kamera i sličnih uređaja najčešće rade na način da integriraju pulsirajuće infracrvene svjetlosne izvore velikom brzinom na konvencionalne prijenosne TV kamere, te povezuju tzv. *time of flight* emitiranih i reflektiranih svjetlosnih zidova kako bi upravljali mjerama dubine scene.

Potreba za većom kvalitetom 3D sadržaja ne može se u potpunosti zadovoljiti trenutačnim pristupima, zbog čega je razvoj novih neizbježan. No, način generiranja 3D sadržaja ne utječe na krajnji rezultat. Svi slučajevi sastoje se od regularnog 2D videozapisa u boji, Europskog digitalnog TV formata (*720 × 576 luminance pels, 25 Hz, interlace*) i iste prostorno-vremenske rezolucije. Svaki od navedenih slika dubine pohranjuju informacije o dubini kao 8-bitne sive vrijednosti s nivoom sive (*eng. greylevel*), gdje nula koja označuje najudaljeniju vrijednost, a vrijednost 255 definira najbližu vrijednost. Prevođenje formatske reprezentacije ove informacije na njene prave, metričke vrijednosti, koje su potrebne za generiranje virtualnog pogleda i fleksibilne na način da poštuju 3D scene s različitim dubinskim karakteristikama, te sive vrijednosti, su normalizirane na dva glavna dubinska isječka ravnina (*eng. depth clipping plane*). Takozvana najbliža ravnina, *near clipping plane* Z_{near} (nivo sive 255) definira najmanju metričku dubinsku vrijednost Z, koja se može prikazati u partikularnoj slici dubine. Nivo sive 0 (nula) definira najdužu metričku dubinsku vrijednost, prikazuje se kao najudaljenija ravnina, *far clipping plane* Z_{far}. U slučaju linearne kvantizacije dubine, sve ostale vrijednosti (1-254) mogu se prikazati formulom:

$$Z = Z_{far} + v \cdot \frac{Z_{near} - Z_{far}}{255} \text{ with } v \in [0,\ldots, 255] , \tag{2.1}$$

gdje *v* specificira vjerodostojnu vrijednost nivoa sive[2].



Slika 2.1 - 2D slika u boji (Europski digitalni TV format) i njen 8-bitna slika dubine s istom prostorno-vremenskom rezolucijom. Prikaz rada i pojašnjenje formule.

Kreirana 3D slika mora proći kroz kodiranje i kompresiju kako bi se mogla napraviti transmisija do krajnjeg korisnika. Kako bi krajnjim korisnicima pružili trodimenzionalni sadržaj, monoskopski videozapis u boji i njegova odgovarajuća informacija moraju biti komprimirani i odaslani preko konvencionalne 2D digitalne TV prijenosne infrastrukture. Kako bi se osigurala kompatibilnost s postojećim 2D TV prijamnicima osnovni videozapis u boji mora biti šifriran (*eng. encoded*) korištenjem standardnog MPEG-2 alata. S druge strane, dopunska slika dubine može biti komprimirana korištenjem bilo kojih novijih, efikasnijih dopuna standarda MPEG standarda, poput MPEG-4 Visual ili AVC (Advanced Video Coding)[3].

Višepogledno video kodiranje (*eng. Multiview video coding*) već je postalo standardizirano kao ekstenzija MPEG-2 standarda, AVC standarda i HEVC standarda. Višepogledna ekstenzija AVC-a je označena kao MVC (Multiview Video Coding) i ekstenzija HEVC-a kao MV-HEVC (Multiview HEVC). Ove višepogledne ekstenzije standardizirane su na način da i alati kodiranja nižeg nivoa su virtualno isti kao za monoskopsko video kodiranje. Nedavno, istraživanja su pokazala da je efikasnost predviđanja unutar slike (*eng. infra-frame*) virtualno ista za MV-HEVC i HEVC uvećan s *Intra Block Copy* alatom, koji je prvenstveno dizajniran za računalno generiran sadržaj[4].

Alati za 3D video kodiranje temeljeni su na sintetiziranom pogledu, predviđanju unutar slike putem 3D mapiranja definiranog dubinom, kodiranja područja MV-HEVC i 3D-HEVC dijele istu sintaksu visokog nivoa tokova bitova (*eng. bitstream*), a više-petlja (*eng. multi-loop*) strukture kodera i dekodera je česta arhitektura korištena u implementacijama[4].

Efikasnost kompresije može se poboljšati uklanjanjem visokofrekvencijskih komponenata iz oba pogleda dubinskih mapa. Svaka slika dijeli se u područja temeljena na njihovim dubinskim vrijednostima. Područja koja su daleko od kamere su filtrirana niskopropusnim filtrom grublje nego područja bliža kameri, što

osigurava da uklanjanje detalja ne oštećuje značajno kvalitetu slike. Područja dalje od kamere se također mogu kvantizirati više nego ona bliža kameri. Ova korelacija može se iskorištavati putem arhitekture skalabilnog video kodiranja (*eng. Scalable Video Coding*), SVC, gdje temeljni sloj kodira poglede i sloj poboljšanja nosi dubinske podatke. Trenutno je ovaj pristup dio 3D-HEVC standarda gdje se polje kretanja dubinskih mapa može predvidjeti iz odgovarajućeg polja kretanja[4].

DIBR tehnika je vrlo popularna ovdje, gdje podaci o dubini i pogled se koriste za generiranje virtualne slike. DIBR tehnika odabrana je od strane MPEG-a kao referentna sinteza modela (*eng. framework*) za arhitekturu videeozapisa sa slobodnom točkom gledišta (*eng. free viewpoint video architectures*), koja ovisi o višepoglednom tekstura-plus-dubina formatu (*eng. multiview texture-plus-depth*)[4].

DVB (Digital Video Broadcast) projekt zahtjeva korištenje MPEG-2 *Systems Layer* specifikacija za distribuciju audio-vizualnih informacija putem kabela (DVB-C), satelita (DVB-S) te zemaljskog (DVB-T) odašiljača. Ovaj standard, zbog visokog, svjetskog prihvaćenja, od velike je važnosti kod svih budućih 3D-TV sistema kao i izgradnje svojih usluga distribucije ovih tehnologija. Projekt je realiziran od strane ATTEST konzorcija.

Transmitirani podaci, za prikaz na 3D TV-u, dekodirani su u 3D-TV prijamniku da bi primili dekomprimirane sekvence videozapisa i pripadajuće u boji (te ostale meta podatke). Iz reprezentacije formata ovih podataka, DIBR algoritam generira virtualne lijeve i desne poglede (*eng. left- and right-eye view*) za trodimenzionalnu reprodukciju stvarne scene na stereoskopskom ili autosteroskopskom, 3D-TV ekranu[3].

3. 3D Izobličenje slike

DIBR (Depth-image-based rendering), je proces sintetiziranja virtualnih pogleda statičnih ili dinamičnih slika u boji i udruženih dubinskih informacija. Izvorne točke spomenutih slika su prvo projiciranje u 3D svijet korištenjem vjerodostojnih informacija o dubini. U spomenutom 3D svijetu, 3D prostorne točke su onda projiciranje u prostornu sliku virtualne kamere, koja mora biti postavljena na točno određen položaj gledanja. Koncentracija, "zgušnjavanje" projekcije 2D u 3D i naknadna projekcija 3D u 2D naziva se 3D izobličenje slike (*eng. Image Warping*)[2].

Jednadžbe koje dobivamo uzimajući dvije kamere i proizvoljni 3D prostor označen *M*, s projekcijama *m* i *m′*. Pod pretpostavkom da je svjetski koordinatni sustav jednak koordinatnom sustavu kamera prve kamere, rezultat jednadžbi perspektivnih projekcija je:

$$\begin{aligned} \tilde{m} &\cong A' P_n \tilde{M} \\ \tilde{m} &\cong A\, P_n D \tilde{M}\,, \end{aligned} \tag{3.1}$$

gdje $\tilde{m}$, $\tilde{m}'$ i respektivni $\tilde{M}$ simboliziraju dvije 2D slikovne točke, poštivajući 3D prostornu točku u homogenoj notaciji a simbol $\cong$ označava jednakost do *non-zero* faktora skaliranja. 4×4 matrica D sadrži rotaciju R i translaciju t koja transformira 3D točku svjetskog koordinatnog sustava u koordinatni sustav kamere drugog pogleda te 3×3 matrice A i A' specificiraju unutrašnje parametre prve kamere, poštivajući drugu. Istovjetna 3×4 matrica P_n određuje takozvanu normiranu perspektivnu projekciju matrice.

Prikazivanje srodne 3D površinske točke *M*, koje i dalje ovisi o dubinskoj vrijednosti Z, dobija se preuređivanjem jednadžbe $\tilde{m} \cong A' P_n \tilde{M}$:

$$M = Z A^{-1} \tilde{m} \tag{3.2}$$

Jednadžba koja definira dubinski ovisnu relaciju između odgovarajućih točaka u dva perspektivna pogleda iste 3D scene dobiva se klasičnom jednadžbom dispariteta (*eng. disparity equation*) nastalom uvrštavanjem prethodnih dviju jednadžbi (3.1),(3.2)

Dobivena jednadžba:

$$Z'\tilde{m}' = ZA'RA^{-1}\tilde{m} + A't \tag{3.3}$$

Navedenom jednadžbom dispariteta (3.3) također se može smatrati formalizam 3D izobličenja slike, koji se može upotrebiti za generaciju proizvoljnog novog pogleda iz već poznatih referentnih slika. Potrebna je definicija pozicije i orijentacija virtualne kamere relativna referentnoj kameri te deklaracija unutarnjih parametara virtualne kamere. Ukoliko su dubinske vrijednosti odgovarajućih 3D prostornih točaka poznate za svaki piksel izvorne slike, virtualni pogled može biti sintetiziran primjenjujući prethodnu jednadžbu na sve izvorne slikovne točke [2].

4. 2D-u-3D Video Konverzija

3D TV sustavi temeljeni su na stereoskopskom ili višepoglednom prikazu sadržaja. Sustav radi na način da spaja najmanje dva prijenosa videozapisa dobivenih od kamera koje su na neznatno drugačijim položajima. Spomenuti videozapisi šalju se na 3D monitor kako bi se kod korisnika stvorila stereoskopska percepcija dubine.

Cilj 2D-u-3D konverzije je generiranje spomenutog drugog (ili višeg) video prijenosa iz originalnog monoskopskog, 2D prijenosa (*eng. stream*)[2].

4.1. Princip konverzije

Budući da 2D videozapis nema dubinu, ravan je, najčešće postavljeno pitanje je kako pretvoriti takav video u trodimenzionalni. Razvojem ljudske percepcije, na gotovo svim 2D slikama postoji interpretacija vizualnih informacija koje dovode do razumijevanja i opažanja dubinskih relacija među objektima na 2D monoskopskim slikama.

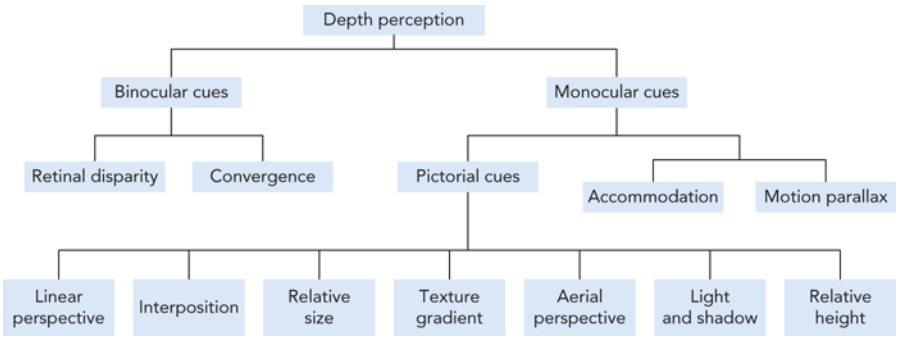
Slika 4.1 - Primjer 2D fotografije gdje se dubina može lako percipirati.

Upravo zbog razvoja našeg perceptivnog sustava i interpretacije te percepcije dubine 2D monoskopskih slika opisanih gore, moguća je konverzija 2D monoskopskih slika u 3D stereoskopske (i višepogledne (*eng. Multi-View*))

videozapise je moguća. Ljudski vizualni sustav, HVS (Human Visual System), koristi i razliku između slika dobivenih iz oba oka za percepciju dubine kao i ostale znakove, naučene vremenom, koji također daju informacije o dubini objekata na sceni. Bez obzira na dvodimenzionalnost slike, ljudsko oko može percipirati dubinu i zaključiti koji se objekt nalazi dalje, a koji bliže, stereoskopska informacija o dubini nije potrebna[3].

Ljudski perceptivni sustav može interpretirati više različitih znakova dubine kako bi generirao reprezentaciju okruženja u 3D svijetu, time dajući volumen dvjema "ravnim" reprezentacijama dobivenim ljudskim očima. Dubinski znakovi ili znakovi dubine (*eng. depth cues*) izrazi su za informacije o dubini u slikama, a razvrstavaju se u binokularne (dobivene od oba oka) ili monokularne (dobivene od jednog oka) dubinske znakove. Binokularni dubinski znakovi koriste manje razlike između slika koje opažaju oba oka kako bi iz njih dobili informaciju o dubini. Monokularni dubinski znakovi pomažu ljudskom perceptivnom sustavu ekstrahirati relativne dubine između objekata u pojedinačnom pogledu scene.

Slika ispod detaljnije prikazuje grananje binokularnih i monokularnih dubinskih znakova:

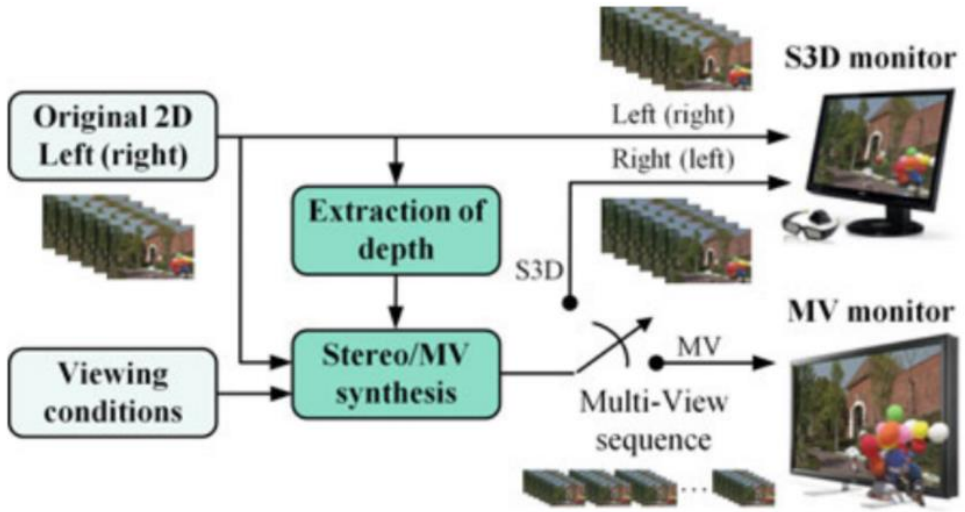


Slika 4.2 - Prikaz svih dubinskih znakova u slikama.

Proces konverzije 2D monoskopskog videa u 3D steroskopski (ili višepoglednog) videozapisa je konverzija, pretvaranje sekvence monoskoptske slike u sekvencu stereoskopske slike ili višepoglednih video isječka. Konverzija zahtjeva ekstrakciju informacije o dubini izvorne monoskopske slike i generiranje nove slike koja će prikazivati scenu iz neznatno pomaknute točke gledišta na način da obje slike (izvorna monoskopska i generirana stereoskopska) formiraju stereoskopski slikovni par. Točnije, pretvorba se ostvaruje ekstrakcijom ili direknim generiranjem dubinskih struktura scene iz analiziranog izvora monoskopske slike. Dobivena informacija

koristi se za generiranje nove slike iz stereoskopskog slikovnog para ili za renderiranje više pogleda ako se radi o višepoglednim slikama[2].

Slikovno objašnjenje konverzije 2D u 3D prikazuje se preko modela. Spomenuti model prikazuje se kao proces generiranja slike na način da se jednom oku prikaže ista slika izvorne 2D scene koju vidi drugo oko. Na taj način, oba oka vide istu scenu iz različitih kutova, što dovodi do zaključka da proces konverzije zahtjeva reproduciranje točke gledišta prvospomenutog oka. Točna ekstrakcija dubine je osnovni dio procesa zbog ovisnosti o dubini objekata na sceni koji će se prikazati na različitim prostornim lokacijama u novonastaloj stvorenoj slici[2].



Slika 4.3 - Prikaz općenitog modela za 2D-u-3D konverziju.

4.2. Monokularni dubinski znakovi u videozapisima

Ukoliko binokularni dubinski znakovi (*eng. binocular depth cues*) nisu dostupni, sve informacije o dubini moraju biti ekstrahirane iz monokularnih dubinskih znakova izvorne slike kako bi 2D-u-3D video konverzija bila moguća. Monokularni dubinski znakovi (*eng. monocular depth cues*) dijele se u dvije glavne kategorije: slikovni (*eng. pictorial cues*) i temeljene na pokretu (*eng. motion-based*).

Motion-based Dubinske informacije temeljene na pokretu (*eng. motion-based depth data*) u video sekvencama prikazuju se kao paralaksi kretanja (*eng. motion parallax*), koji se mogu definirati kao razlika u prostornoj poziciji objekata kroz vrijeme. Kada se objekti pomiču u sceni, njihova prostorna pozicija (*eng. spatial position*) u slici mijenja se ovisno o lokaciji tih objekata u sceni. Kretanje objekta izazvano je kretanjem kamere i/ili objekta. Kod percepcije paralaksa kretanja, objekti koji se nalaze dalje u sceni daju dojam da se sporije pomiču, za razliku od objekata koji se nalaze bliže - bliži objekti kreću se brže. Ljudski vizualni sustav razliku u brzini pomicanja objekata koji su zapravo statički, prevodi kao informaciju o dubini. Navedeni paralaks kretanja jak je i bitan dubinski znak koji se može iskoristiti za izdvajanje dubinskih informacija potrebnih za 2D-u-3D konverziju.

Pored motion-based informacija o dubini koje se mogu dobiti i iskoristiti za 2D-u-3D video konverziju, postoji nekoliko slikovnih dubinskih znakova (*eng. pictorial depth cues*) u monoskopskim statičkim slikama koji se mogu iskoristiti u procesu dubinske ekstrakcije za navedenu pretvorbu. Potrebno je istaknuti da ljudski perceptivni sustav može kreirati stabilnu volumetrijsku reprezentaciju scene iz svih dubinskih znakova u monoskopskim scenama.

Interpozicija objekata jedan je od glavnih slikovnih dubinskih znakova koje koristi ljudski perceptivni sustav. Kada je jedan objekt ispred drugog, objekt bliži promatraču prepriječit će prikaz objekta koji se nalazi dalje. Ovaj dubinski znak daje informaciju o dubinskom poretku objekata, ali ne i magnitudi.

Sljedeći bitni dubinski znak je vezan za ograničenje dubinskog polja kamera. Spomenuto ograničenje uočava se se na način da su neki objekti u fokusu, dok drugi nisu. Ljudski perceptivni sustav koristi mehaniku akomodacije oka kako bi zadržali željene objekte u fokusu. Navedeni mehanizam koristi se i kod kamera, reproduciran je u video sekvencama zbog spomenutih ograničenja dubinskih polja. U 2D slici, željeni objekti, objekti u interesu bit će u fokusu, dok su ostali objekti izvan fokusa (izgledaju “mutno”). Razlika između objekata u fokusu i izvan fokusa daje informacije koje određuju dubinski odnos između tih objekata[2][3].

Slika 4.4 - Prikaz dubinskog polja na 2D slici (isječak iz filma Marvel's Avengers - Infinity War, 2018.).

Ostali dubinski znakovi vezani su uz našu interpretaciju vizualne informacije. Indikatori koliko daleko ili blizu su locirani objekti u sceni su varijacije u veličini, teksturi, svijetlu, boji te druge karakteristike objekata na slici. Analiziranjem tih karakteristika i varijacija u slici moguće je odrediti relativnu dubinsku poziciju tih objekata. Glavna poteškoća vezana za korištenje ovih dubinskih znakova u računalnim algoritmima je da su sve ove karakteristike naučene i vrlo teško izdvojene iz slika automatski - potrebno je prilagođavanje algoritama.

Statičke slike kao i sekvence slika u pokretu sadrže širok opseg dubinskih informacija koje se mogu izvući i kreirati S3D verziju 2D slike. Izdvajanje podataka i informacija o dubini preciznija je i pouzdanija ukoliko postoji ljudska intervencija u algoritam.

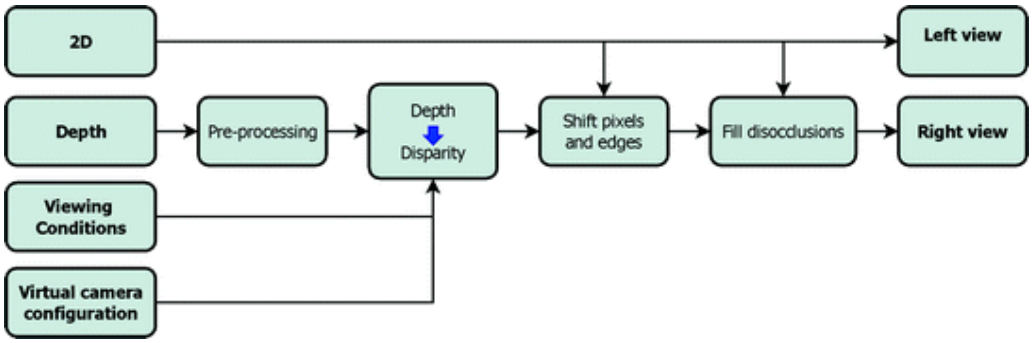
4.3. DIBR model za konverziju 2D-u-3D

Proces se sastoji od korištenja informacija iz piksela (*eng. per-pixel information*) o dubini za pozicioniranje svakog piksela originalne, izvorne slike odgovarajućoj prostornoj lokaciji (*eng. spatial location*) u novogeneriranoj slici. Postoje tri osnovne operacije u ovome procesu, Slika 6:

-pretvorba informacije o dubini u disparitetne vrijednosti kao horizontalne razlike između odgovarajućih piksela u originalnim i renderiranim slikama, za svaki piksel u originalnoj slici temeljen na specifičnom geometrijskom pogledu.

-premještanje (*eng. shifting*) piksela originalne slike u nove pozicije u novogeneriranoj slici, ovisno o disparitetnoj vrijednosti (*eng. disparity*).

-popunjavanje mjesta gdje su se pojavile disokluzije kao posljedica premještanja piksela u prethodnoj operaciji.



Slika 4.5 – Prikaz 3 operacije DIBR procesa

Proces je kontroliran od strane skupa parametara ovisnih o stanju gledanja (*eng. viewing conditions*). U DIBR modelu, informacija o dubini prikazana je kao skup 2D podataka u kojemu vrijednost svakog piksela pruža dubinu.

Imajući na oku prethodnu sliku (Slika 6.), prvi korak kod ovog procesa je određivanje konfiguracije virtualne kamere, točnije, gdje pozicionirati virtualne kamere kako bi izgenerirali stereoskopski sadržaj. Relativna pozicija kamera, zajedno sa stanjima gledanja, određuju parametre za pretvorbu informacija o dubini u disparetete između dvaju slika - ovaj proces kreira mapu dispariteta. Koriste se paralelne kamere. Može se koristiti originalni 2D sadržaj kao jedna kamera, te pozicionirati drugu blizu prvoj kako bi se formirao stereo par. Drugi način je korištenje originalnog 2D sadržaja kao centralni pogled i generiranje dva nova pogleda, jedan sa svake strane originalnog [2].

Generiranje samo jednog *single-view* toka slika je jednostavnije i manje računalno zahtjevno te osigurava da barem jedan pogled (koji se sastoji od originalnih slika) ima veću kvalitetu.

Generiranje slika za dva pogleda ima prednosti - obje generirane slike su bliže originalnoj, te kao posljedice, disokluzije ili novoizložene regije su manje i raširene na dvije slike.

Postoje tri glavna parametra koja se moraju uzeti u obzir kod pretvorbe informacija o dubini u disparitetne vrijednosti korištene u DIBR procesu za kreiranje novih slika koje formiraju stereoskopski par. Navedeni parametri su: udaljenost od zaslona, širina zaslona i korisnikova interokularna udaljenost.

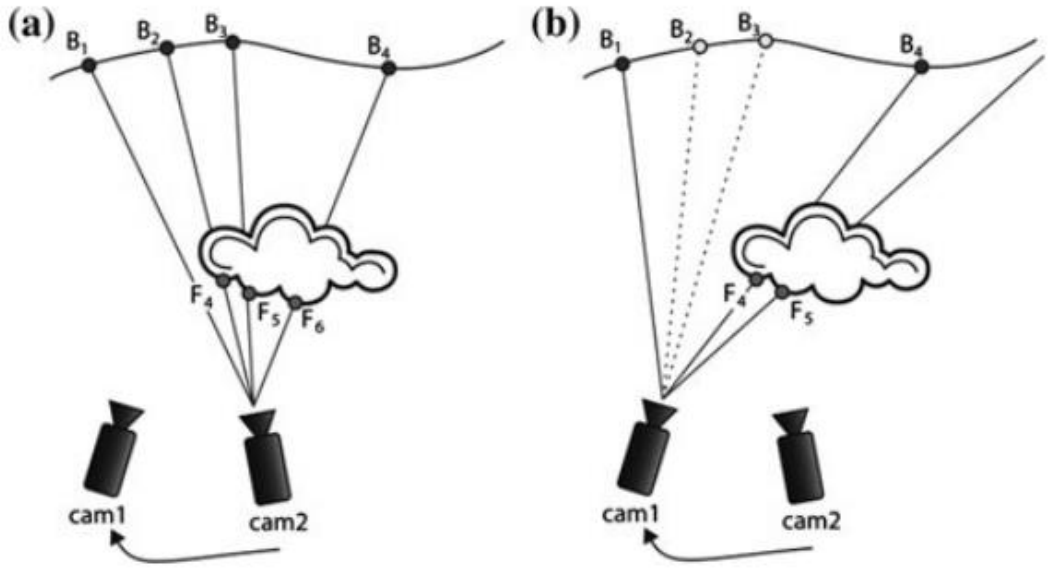
Druga glavna operacija u DIBR procesu je, ranije spomenuto, premještanje piksela na njihove pozicije u novoj slici. Originalne dubinske vrijednosti su predprocesirane kako bi uklonile šum i smanjile broj manjih disokluzija na renderiranoj slici. Nakon toga, dubinske vrijednosti pretvaraju se u vrijednosti dispariteta uzimajući u obzir uvjete pogleda (*eng. viewing conditions*).

Ovaj proces može se teoretski interpretirati kao projekcija 3D točaka na slikovno područje (*eng. image plane*) nove virtualne kamere. Točke iz originalne slike kod kojih je dubina dostupna su pozicionirane u 3D prostoru i tada ponovno projicirane na slikovno područje nove virtualne kamere. U praksi, premještanje je horizontalno zbog paralelne konfiguracije para kamera, te su pikseli horizontalno premješteni na njihove nove pozicije bez potrebe traženja njihovih stvarnih pozicija u 3D prostoru. Navedeno horizontalno premještanje piksela je vođeno disparitetima koji su izračunati korištenjem konfiguracija kamere te pravilom da objekti ispred moraju zakloniti objekte iza.

Zadnja glavna operacija u DIBR procesu je popunjavanje “rupa” kreiranih oštrim rubovima u disparitetnoj mapi. Bilo kada je piksel s većim disparitetom popraćen pikselom s manjim disparitetom, razlika u tim disparitetima, kao posljedicu, stvara ili okluziju (superpozicija objekata) ili disokluziju (nedostatak vizualne informacije) [1][2].

5. Popunjavanje rupa u sintetiziranom pogledu (popunjavanje disokluzija)

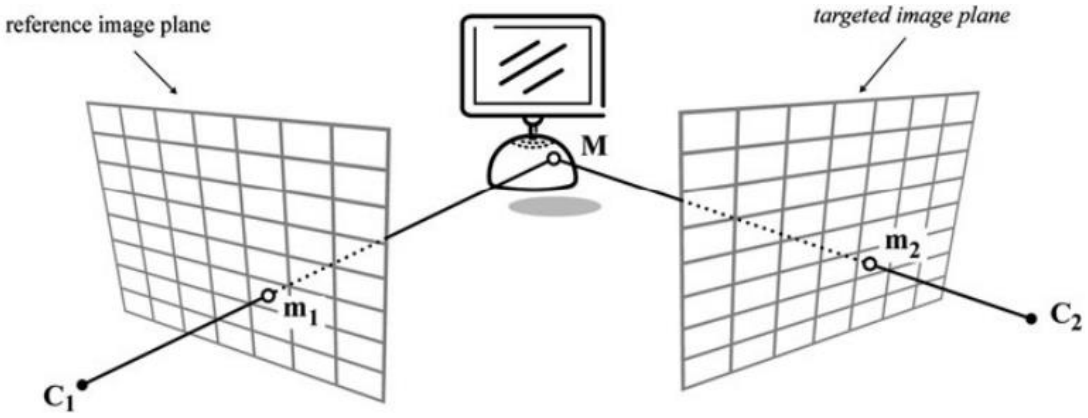
Reprezentacija 3D video podataka i njihovih višepoglednih ekstenzija poznati su kao tekstura-plus-dubina ili, za višepogledne, višepogledna tekstura-plus-dubina (MVD). Pružaju obogaćenje običnih 2D videozapisa njihovim udruženim dubinskim podacima. 2D videozapis pruža informacije o teksturi, intenzitet boje, a videozapis s dubinom prikazuje
Z- udaljenost po pikselu između kamere i 3D točke u vizualnoj sceni.



Slika 5.1 - Stereo konfiguracija gdje se svi pikseli ne mogu vidjeti iz svih pogleda kamera.

Okluzije i disokluzije spomenute su ranijim poglavljima, a slika iznad prikazuje kako kod prebacivanje pogleda cam2 u cam1, točke B2 i B3 su u primjeru a) zaklonjene u cam2, a u primjeru b) nastaje disokluzija kod cam1. Također, kao što je ranije navedeno, što je razmak između kamera veći, to su veće i disokluzije.

Novi sintetizirani pogled uključuje funkciju mapiranja točaka iz jednog pogleda (referentni slikovni prostor) u drugi (ciljani slikovni prostor). Prije nego što dođemo do navedene funkcije, treba napomenuti nekoliko bitnih stavki:
Intenzitet referentne pogledne slike I_1 na koordinatama piksela (u_1, v_1) je označen s $I_1(u_1, v_1)$. *Pinhole* model kamere je korišten za projekciju I_1 u drugi pogled $I_2(u_2, v_2)$ s danim podacima o dubini $Z(u_1, v_1)$.



Slika 5.2 - Prikaz 3D image warping.

3D image warping proces opisan je u ranijem poglavlju gdje je spomenuta mogućnost sintetiziranja novog virtualnog pogleda. Potrebna je definicija pozicije i orijentacija virtualne kamere relativna referentnoj kameri te deklaracija unutarnjih parametara virtualne kamere [2].

5.1. Proces 3D izobličenja slike kod sinteze

3D izobličenje slike se dijeli u dva koraka: prva projekcija (*eng. backprojection*) referentne slike u 3D svijet popraćen ponovnom projekcijom 3D scene na ciljanu površinu slike (*eng. image plane*). Ako pogledamo na lokaciju piksela (*eng. pixel location*) (u_1, v_1), prvo, prva projekcija je po pikselu izvedena iz površine slike I_1 2D referentne kamere na koordinate 3D svijeta. Nakon toga, druga projekcija izvedena je iz 3D svijeta na površinu slike I_2ciljane kamere na lokaciji piksela (u_2, v_2), isto vrijedi i za svaku lokaciju sljedećeg piksela. Potrebna su tri parametra za izvedbu ovih operacija: K_1, R_1 i $t_1$koji označuju 3×3 unutrašnju matricu, 3×3 ortogonalnu rotacijsku matricu i 3×1 translacijski vektor referentnog pogleda I_1. Ponovno projicirana točka 3D svijeta $\mathbf{M} = (x, y, z)^T$ izražena je pomoću nehomogenih (*eng. non-homogeneous*) koordinata kao:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = R_1^{-1} K_1^{-1} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \lambda_1 - R_1^{-1} t_1 \tag{5.1}$$

gdje je λ_1 pozitivni faktor skaliranja.

Gledajući na parametre ciljane kamere, K_2 , R_2 i t_2 ponovno projicirane točke 3D svijeta **M** = $(x, y, z)^T$ su tada mapirane u koordinate ciljane 2D slike $(u', v', 1)^T$ homogenih koordinata kao:

$$\begin{pmatrix} u_2' \\ v_2' \\ w_2' \end{pmatrix} = K_2 R_2 \begin{pmatrix} x \\ y \\ z \end{pmatrix} + K_2 t_2 \tag{5.2}$$

Ciljane koordinate mogu se izraziti funkcijom referentnih koordinata:

$$\begin{pmatrix} u_2' \\ v_2' \\ w_2' \end{pmatrix} = K_2 R_2 R_1^{-1} K_1^{-1} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \lambda_1 - K_2 R_2 R_1^{-1} t_1 + K_2 t_2 \tag{5.3}$$

Uobičajeno se dodaje svjetski koordinatni sustav sustavu prve kamere tako da je $R_1 = I_3$ i $t_3 = O_3$, iz čega nastaje:

$$\begin{pmatrix} u_2' \\ v_2' \\ w_2' \end{pmatrix} = K_2 R_2 K_1^{-1} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \lambda_1 + K_2 t_2 \tag{5.4}$$

Gdje $(u_2', v_2', w_2')^T$ predstavlja homogene koordinate točke 2D slike $\boldsymbol{m_2}$, i pozitivni faktor skaliranja λ_1 jednak je:

$$\lambda_1 = \frac{z}{c} \; gdje\; je \begin{pmatrix} a \\ b \\ c \end{pmatrix} = K_1^{-1} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} \tag{5.5}$$

U zadnjem koraku, homogeni rezultat konvertiran je u lokaciju piksela (*eng. pixel location*) kao:

$$(u_2 v_2) = \left(\frac{u_2'}{w_2'}, \frac{v_2'}{w_2'}\right) \tag{5.6}$$

$\mathcal{Z}$ je treći komponent točke 3D svijeta **M**, što pokazuje informaciju o dubini na lokaciji piksela (u_1, v_1) slike I_1. Ovi podaci smatraju se ključnim sporednim / ”bočnim” informacijama (*eng. side information*) za povratak odgovarajućih lokacija piksela na drugoj slici I_2.

5.2. Disokluzije i problemi

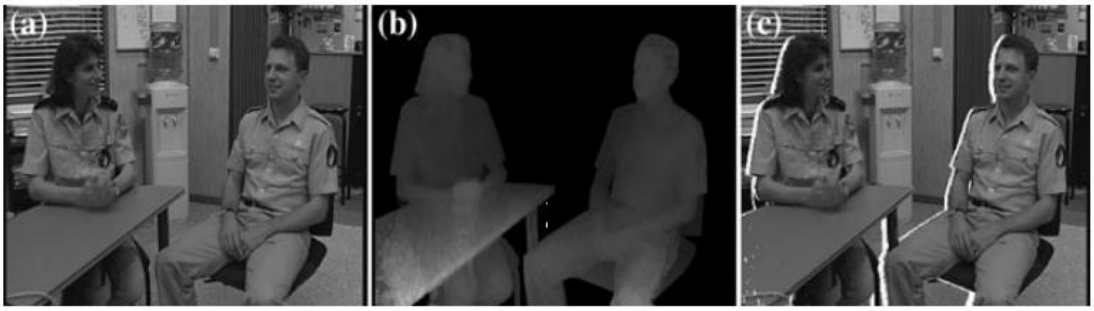
Inherentni problem iznad opisanog algoritma 3D izobličenja slike je to da baš savki piksel ne postoji u oba pogleda. Zbog oštrih diskonuiteta u podacima o dubini, ovaj algoritam može izložiti područja scene koja su koja su okludirana (*eng. occluded*) u referentnom pogledu i tako postati vidljiva u sintetiziranom pogledu. Postoje brojna rješenja za ovaj problem koja uglavnom prate dvije općenite linije istraživanja: dubinsko predprocesiranje (*eng. depth pre-processing*) i popunjavanje rupa (*eng. hole filling*). Predprocesiranje se odvija prije DIBR-a, a sintetiziranje nakon DIBR-a.

Predprocesiranje dubinskog videozapisa dozvoljava redukciju broja i veličine disokludiranih područja, na način da izglađuje dubinske diskonuitete, što se uglavnom radi pomoću Gaussian filtra. Izglađivanje čitavog dubinskog videozapisa pravi dosta veću štetu nego primjena korekcija oko rubova, što je dovelo do predlaganja brojnih novih adaptivnih filtera kako bi se reducirale i disokluzije i izobličenja izazvane filtrima u dubinskom videozapisu. Bilateralni filter, svojim štićenjem rubova (*eng. edge-preserving*), koristi se za povećavanje dubinskog videozapisa. U usporedbi s filtrima temeljenim na Gaussian-ovom filtru, bilateralni filter vrši operacije i u prostoru (u smislu prostorne blizine piksela, jednako kao Gaussov filter) i u fotometrijskom smislu (u smislu blizine intenziteta boje ili svjetline piksela) (*eng. spatial space*) i u prostoru intenziteta boje (*eng. color intensity space*), što rezultira u boljem očuvanju oštrih dubinskih promjena u vezi s varijacijom intenziteta u prostoru boje, i u konzistentnim granicama između teksture i dubinske slike.

Veće disokluzije mogu ostati i nakon dubinskog pred procesiranja, zbog čega je potrebna interpolacija nedostajućih vrijednosti. Koristi se prosječni filter (Average filter). Average filter ne čuva informacije o rubovima interpoliranog područja, što je

razlog pojavljivanja artefakata u visoko teksturiranim područjima. Mark i Zhan-wei predlažu spajanje dvaju izobličenih slika dobivenih od dvaju različitih prostornih ili privremenih gledišta (*eng. viewpoint*), gdje su pikseli u referentnoj slici procesirani u okulzijsko-kompatibilnom redoslijedu. S druge strane, Tauber predlaže popravljanje slika (*eng. image inpainting*) i tvrdi da može pružiti efikasni model za popunjavanje disokluzija teksturama i širenjem struktura te da bi trebao biti integriran s *image-based rendering* (IBR) tehnikama[2][3][5][8].

Iako ove metode pružaju uglavnom efikasan način popunjavanja nastalih disokluzija, i dalje postoje problemi poput degradacije područja bez disokluzija u dubinskoj mapi, dubinski inducirana distorzija izobličenih slika te neželjenih artefakata u *inpainted* disokluziranim područjima. Predlaže se, kao rješenje, adaptivno dubinsko mapno predprocesiranje koje uglavnom vrši operacije na rubovima, i postprocesiranje temeljeno na impainting-u koje koristi informacije o dubini u slučaju velikog razmaka među kamerama.

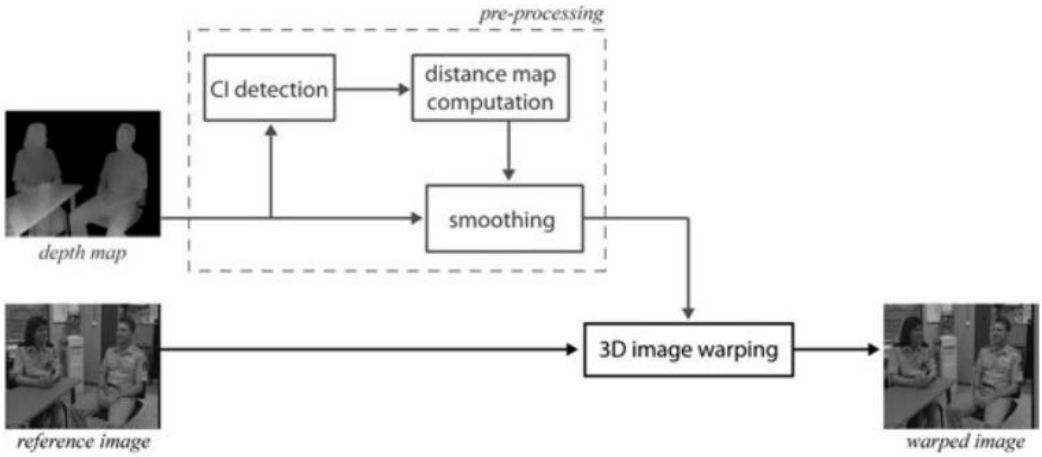


Slika 5.3 - Prikaz DIBR izobličene slike, gdje je a) referentna tekstura, b) referentna dubina, c) sintetizirani pogled.

5.3. Predprocesiranje dubinskog videa

Navedeno poglavlje bavi se problemom pojave i popunjavanja disokluzija kod kamera s manjom međusobnom udaljenošću. Rezultat manje udaljenosti između kamera je otkrivanje manjih disokluzija. Kao što je spomenuto u ranijem poglavlju, jedan od načina rješavanja navedenog problema je „izglađivanje“ dubinskih diskonuiteta. Predloženo je korištenje adaptivnog filtera, umjesto izglađivanja čitavog dubinskog videozapisa, koji uzima u obzir udaljenost do rubova. Prvi korak je primjena stadija pred-procesiranja kako bi se izdvojili rubovi dubinske mape, koji su sposobni razotkriti područja disokluzija.

Spomenuti rubovi navode se kao CI u daljnjem tekstu (*Contours of Interest*). Ova prostorna informacija omogućava izračun informacije o duljini te, nakon toga, zaključivanje informacije o težini kod odabira operacije filtriranja [2].
Filteri koji će detaljnije biti opisani su Gaussian i Bilateralni filter.



Slika 5.4 - Prikaz predprocesiranja dubine prije DIBR-a

5.3.1. Gaussian Filter

Gaussian filter modificira ulaznu dubinsku sliku Z konvulacijom s razdvojnom Gaussian g_{2D} funkcijom

$$(Z * g_{2D})(u,v) = \sum_{x=\frac{-w}{2}}^{\frac{w}{2}} \sum_{y=\frac{-h}{2}}^{\frac{h}{2}} Z(u-x, v-y) \cdot g_{2D}(x,y)\,, \tag{5.7}$$

gdje dvodimenzionalna približnost razdvojne Gaussian g_{2D} funkcije je razdvojena na x i y komponente, prikazane kao:

$$g_{2D}(x,y) = g_{1D(x)} \cdot g_{1D(y)} = \frac{1}{\sqrt{2\pi\sigma_x}} e^{-\frac{x^2}{2\sigma_x^2}} \cdot \frac{1}{\sqrt{2\pi\sigma_y}} e^{-\frac{y^2}{2\sigma_y^2}}, \tag{5.8}$$

gdje g_{1D} je jednodimenzionalna razdvojna približnost Gaussian funkcije. Parametar w predstavlja dužinu, a h visinu jezgre konvolucije. Parametri σ_x , σ_y su standardne devijacije Gaussian distribucije u horizontalnom i vertikalom smjeru [2].

5.3.2. Bilateralni Filter

Bilateralni filter je drugi „zaglađujući“ (*eng. smoothing*) filter koji čuva rubove (*eng. edge-preserving*). Uz konvoluciju u prostornoj domeni, ovaj filter također vrši operacije u domeni intenziteta. Za razliku od Gaussian filtera, bileteralni filter zamjenjuje vrijednost piksela prosječnom težinom susjeda u domeni prostora i intenziteta. Kao posljedica, oštri rubovi očuvani su sustavnim isključivanjem piksela preko diskonuiteta iz razmatranja. Nova dubinska vrijednost u filtriranoj dubinskoj mapi $\tilde{Z}$ na lokaciji piksela *s = (u,v)* je definirana s:

$$\tilde{Z}_s = \frac{1}{k(\boldsymbol{s})} \cdot \sum_{p \in \Omega} f(\boldsymbol{p} - \boldsymbol{s}) \cdot g\big(Z_p - Z_s\big) \cdot Z_p, \tag{5.9}$$

gdje Ω predstavlja susjedstvo oko ***s*** ispod jezgre konvolucije, a ***k(s)*** je normizacijski član:

$$k(\boldsymbol{s}) = \sum_{p \in \Omega} f(\boldsymbol{p} - \boldsymbol{s}) g(Z_p - Z_s) \tag{5.10}$$

Gaussian razdvojna funkcija (5.7) koristi se u praksi za prostorni filter *f* u prostornoj domeni i dometni filter *g* u domeni intenziteta:

$$\begin{aligned} f(\boldsymbol{p} - \boldsymbol{s}) &= e - \frac{d(p-s)^2}{2\sigma_d^2}, \\ d(\boldsymbol{p} - \boldsymbol{s}) &= \|\boldsymbol{p} - \boldsymbol{s}\|_2 \end{aligned} \tag{5.11}$$

$\|.\|_2$ je Euklidska distanca i :

$$\begin{aligned} g(Z_p - Z_s) &= e - \frac{\delta(Z_p - Z_s)^2}{2\sigma_r^2}, \\ \delta(Z_p - Z_s) &= |Z_p - Z_s| \end{aligned} \tag{5.12}$$

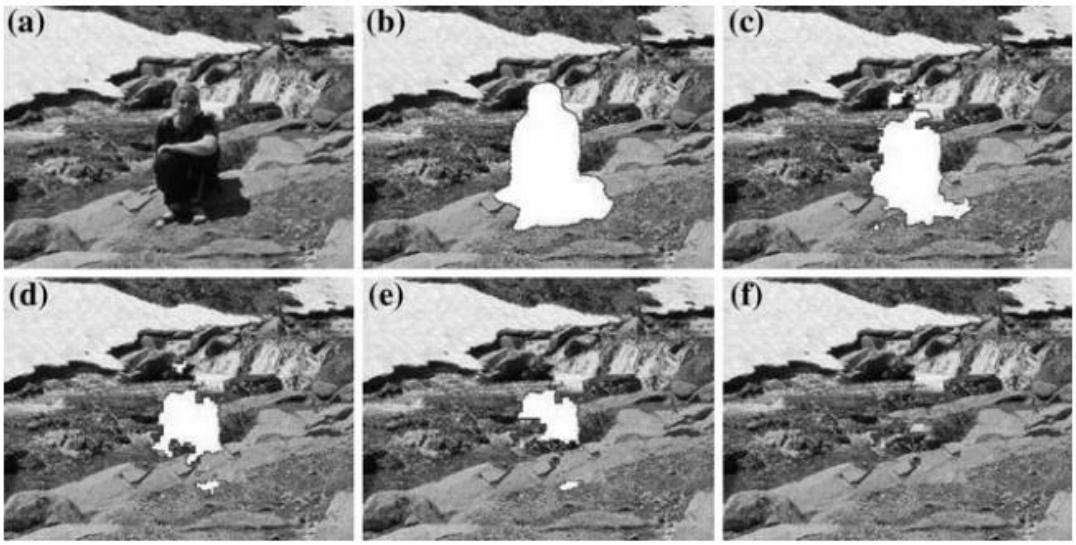
Gdje su σ_d i σ_r standardne devijacije prostornog filtera f i filtra inteziteta *g*. Opseg filtera je tada kontroliran s ova dva ulazna parametra. Bilateralni filter može se smatrati produktom dvaju Gaussian filtera, gdje vrijednost na lokaciji piksela s je izračunata s težinskim prosjekom svojih susjeda prostornom komponentom koja ovisi o udaljenosti piksela, i komponentom ovisnom o intenzitetu, koja sankcionira piksele s drugim intenzitetom [2].

5.4. Popunjavanje slika (*eng. Image Inpainting*)

Image inpainting, također poznat kao *image completion*, je metoda koja se većinom koristi za popunjavanje većih disokluzija. Veći razmak između kamera koji odgovara Free Viewpoint Videozapisu (FVV), za razliku od stereoskopskog, posljedica je pojave većih disokluzija.

Image inpainting radi na način da popunjava piksele u većoj „nepoznatoj" / nepostojećoj regiji slike koristeći informacije iz okruženja („poznate" regije slike). *Image inpainting* koristi se kod većeg broja stvari poput uklanjanja prekrivenog teksta ili logotipa, restoracije oštećenih slika, kompresije slika, te i kreiranje artističkih efekata.

Image inpainting metode široko se klasificiraju kao strukturalni i teksturalni *image inpainting*. Strukturalni image inpainting vrši rekonstrukciju koristeći prijašnje pretpostavke o glatkoći struktura u nedostajućim regijama i graničnim uvjetima. Teksturalni *image inpainting* koristi samo poznate podatke iz teksturalnih primjeraka ili drugih predložaka iz postojećih, poznatih regija. Postoje i drugi načini *image inpaintinga* te njihove kombinacije. Dalje u tekstu opisat će se Criminisijev *Image Inpanting* algoritam, koji kombinira strukturu u teksturalni *image inpainting*, gdje je tekstura neobojena u smjeru jednakog intenziteta prema svojoj snazi [1][7].

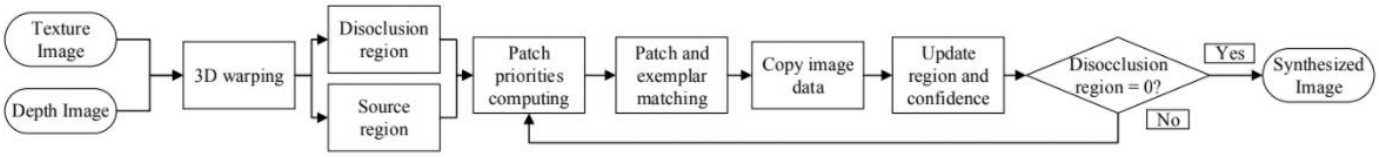


Slika 5.5 - Prikaz uklanjanja dijela slike te popunjavanje veće praznine koristeći Criminsijev algoritam.

5.4.1. Criminsijev algoritam

Criminisi je predložio novi tip algoritma za image inpainting. Algoritam upotrebljava tehniku za sitetinziranje teksture temeljenu na primjercima (*eng. exemplar-based texture sythesis*). Korišten je koncept uzorkovanja iz Elfrosovog i Leungovog pristupa, što je demonstriralo da kvaliteta izlazne sinteze slike ovisi o redoslijedu kojim ide proces *image inpainting*-a.

„Povjerenje“ (*eng. confidence*) se koristi za izračun prioriteta zakrpa (*eng. patch priority*) i za optimizaciju redoslijeda popunjavanja regija od značaja ovisno o njihovim prioritetima. Vrijednosti boje izračunavaju se korištenjem sinteze teksture temeljene na primjercima. Nakon popunjavanja ciljane zakrpe (*eng. target patch*) novim vrijednostima, *confidence* parametar je obnovljen. *Confidence* u sitetiziranoj vrijednosti piksela širi se na način sličan širenju informacija u popunjavanju slike. Kako popunjavanje napreduje, *confidence* vrijednosti opadaju što pokazuje da su vrijednosti boje piksela manje pouzdane u blizini centra ciljane regije.



Slika 5.6 - Prikaz Criminisijevog dijagrama

Prije prikaza dobivenih formula bitno je spomenuti nekoliko notacija. Uzimajući ulaznu sliku I i nedostajeću regiju (*eng. missing region*) Ω, izvorna regija Φ je definirana kao $\Phi = I - \Omega$. Criminisijev algoritam sastoji se od korištenja najbolje prve strategije popunjavanja (*eng. best-first filling strategy*) koja u potpunosti ovisi o prioritetnim vrijednostima koje su dodjeljene svakoj zakrpi na granici $\delta\Omega$. Uzimajući da je zakrpa Ψ_p centrirana na mjestu točke (*eng. point location*) $p \in \delta\Omega$, prioritetna zakrpa P(**p**) je definirana kao produkt dva člana:

$$P(\boldsymbol{p}) = C(\boldsymbol{p}) \cdot D(\boldsymbol{p}) \tag{5.13}$$

gdje je C(**p**) ranije spomenuti *confidence* član koji pokazuje pouzdanost trenutne zakrpe, i D(**p**) je *data* član koji daje posebni prioritet smjeru jednakog intenziteta.

Navedeni članovi (5.13) su definirani kao:

$$C(\boldsymbol{p}) = \frac{1}{|\Psi_p|} \sum_{q \in \Psi_p \cap \Phi} C(\boldsymbol{p}) \tag{5.14}$$

$$D(\boldsymbol{p}) = \frac{\nabla^{\perp} I_p, n_p}{\alpha} \tag{5.15}$$

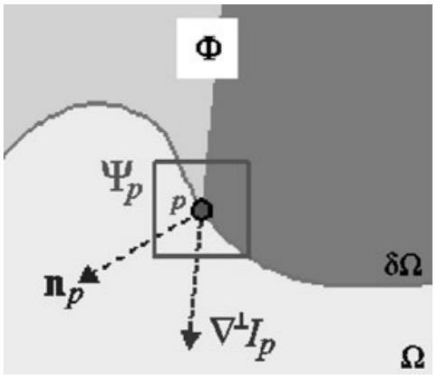
gdje je $|\Psi_p|$ područje od Ψ_p (u smislu broja piksela unutar zakrpe Ψ_p), α predstavlja faktor normizacije (npr. α = 255 za tipičnu sliku svjetline). $\boldsymbol{n_p}$ je jedinični vektor ortogonalan δΩ na točki **p**, i $\nabla^{\perp} = -\partial_y, \partial_x$ predstavlja pravac jednakog intenziteta. C(**q**) predstavlja postotak poznatih (*eng. non-missing*) piksela u Ω, i C(**q**) = 1 u ostalim područjima (regijama).

Kada su svi prioriteti u δΩ izračunati, algoritam koji spaja podudarajuće kocke (*eng. block-matching algorithm*) izvodi najbolji primjerak $\Psi_{\hat{q}}$ da popuni nedostajuće piksele ispod najprioritetnije zakrpe $\Psi_{\hat{p}}$, prijašnje selektirane, kao:

$$\Psi_{\hat{q}} = \arg min_{\Psi \in \Phi} \left\{ d\left(\Psi_{\hat{p}}, \Psi_q\right) \right\} \tag{5.16}$$

gdje je *d*(. , .) distanca između dvije zakrpe, definirana kao suma kvadrata razlika (SSD – *sum of squared differences*). Nakon pronalaska optimalnog izvornog primjerka $\Psi_{\hat{q}}$, vrijednost svakog piksela kojega treba popuniti (*eng. pixel-to-be-filled*) $\hat{p} \in \Psi_{\hat{p}} \cap \Phi$ je kopiran od svog odgovarajućeg piksela u $\Psi_{\hat{q}}$. [1]
Nakon što je zakrpa $\Psi_{\hat{p}}$ popunjena, *confidence* član *C(**p**)* je ažuriran na način:

$$C(\boldsymbol{p}) = C(\hat{\boldsymbol{p}}), \forall \boldsymbol{p} \in \Psi_{\hat{p}} \cap \Phi \tag{5.17}$$



Slika 5.7 - Prikaz dijagrama notacije

5.4.2. View Synthesis Reference Software (VSRS)

Tanimoto je predložio DIBR metodu koja je usvojena od strane MPEG 3D video grupe, poznate kao View Synthesis Reference Software (VSRS). Dubinski diskontinuitet pogrešaka su rješeni izvođenjem naknadnim filtrima nad projiciranu dubinsku mapu. Nakon spomenutog postupka slijedi metoda popravljana ili *inpainting* metoda, koja se koristi za popunjavanje rupa u područjima disokluzija. Metoda se koristi primarno kod sinteze između više pogleda (*eng. inter-viewed sythesis application*), za popunjavanje manjih rupa. Metoda se također može koristiti u slučajevima temeljenim na jednom pogledu (*eng. single view*)[1][8].

Slika 5.8 - Prikaz VSRS modela

6. Praktični rad

Praktični rad prikazuje pet slika iz različitih pogleda, njihove dubinske mape te maske korištene kod tri algoritma. Napisan je manji program koji učitava originalne slike te njihove dubinske mape, i omogućava korisniku pomakom miša izobličavanje slika, tj. 3D dojam. Zbog promjene kuta gledanja kod pokreta miša, nastaju disokluzije koje će biti popravljene s tri različita algoritma. Prvi od algoritama je transformacija udaljenosti (*eng. distance transform*), koji popunjava disokluzije s najbližim postojećim pikselima, te potom koristi Gaussian filter da ugladi prijelaz na postojeće piksele. Program sprema tri slike, od kojih je jedna maska, duga prikazuje sliku s disokluzijama, a treća je krajnja slika (nakon uklanjanja disokluzija) – rezultat Gaussian algoritma.

Slike s disokluzijama i njihove maske proći će kroz još dva algoritma te dalje u radu biti uspoređena.

6.1. Program i kod

Programski dio odrađen je pomoću Microsoft Visual Studia (verzija 2019.) i OpenCV-a. Program učitava sliku koja se pomicanjem miša izobličava, tipkom „F“ spremaju se tri slike – maska, slika s disokluzija i popravljena slika. Tipkom „N“ program učitava sljedeću sliku, dok tipka „C“ zatvara program.

„Programski dio“

```
#include <opencv2/objdetect.hpp>
#include <opencv2/highgui.hpp>
#include <opencv2/imgproc.hpp>
//#include <opencv2/ximgproc.hpp>
#include "opencv2/photo.hpp"

#include <iostream>
#include <queue>
#include <cstdio>
#include <windows.h>

#define _USE_MATH_DEFINES
#include <math.h>
#include <ctime>
#include "wtypes.h"

//#define ATAN_APPROX(X) ( (X)-
1./3*(X)*(X)*(X)+1./5*(X)*(X)*(X)*(X)*(X))
//#define TAN_APPROX(X) (
```

```
(X)+1./3*(X)*(X)*(X)+2./15*(X)*(X)*(X)*(X)*(X))


using namespace cv;
using namespace std;
using namespace std::chrono;
//using namespace cv::ximgproc;

/* Attempt at supporting openCV version 4.0.1 or higher */
#if CV_MAJOR_VERSION >= 4
#define CV_WINDOW_NORMAL                cv::WINDOW_NORMAL
BGR2YCrCb                       cv::COLOR_BGR2YCrCb
#define CV_HAAR_SCALE_IMAGE
cv::CASCADE_SCALE_IMAGE
#endif

/** Function Headers */
void detectAndDisplay(vector<String> fn, vector<String>
fn_depth, cv::Mat image_input, cv::Mat image_depth, int
horizontal, int vertical);
void GetDesktopResolution(int& horizontal, int& vertical);

//static functions
static double rad2Deg(double rad) { return rad * (180 / M_PI);
}//Convert radians to degrees
static double deg2Rad(double deg) { return deg * (M_PI / 180);
}//Convert degrees to radians

Mat Project3D(Mat &image, Mat &depthmap, double &angleX,
double &angleY);


/** Global variables */
cv::String face_cascade_name =
"haarcascade_frontalface_alt2.xml";
//cv::String face_cascade_name = "lbpcascade_frontalface.xml";
cv::CascadeClassifier face_cascade;
std::string main_window_name = "Capture - Face detection";

std::string advertise_window_name = "Image of Advertisement";

cv::Mat debugImage, edited;
cv::Mat mask, result2;


int check_x = -1;
int check_y = -1;

double LEFT = 0.03;
double TOP = 0.03;

int FRAME_SIZE_WIDTH = 320;
```

```
//camera parameters
double h = 300.0; // camera offset
double D = 1000; // depth of image plane
double L = 25000; // focal length of camera
double constant_x = 30; double constant_y = 30;

//element for dilation
Mat element = getStructuringElement(MORPH_ELLIPSE,
      Size(2 * 2 + 1, 2 * 2 + 1),
      Point(2, 2));
```

Definiranje varijabli, opcije zaglavlja, parametri i elementi.

```
#include "main.h"


/**
 * @function main
 */
int main(int argc, const char** argv) {
      cv::Mat frame;
      clock_t start, end;
      int horizontal = 0;
      int vertical = 0;
```

Početni dio

```
cv::namedWindow(advertise_window_name, WINDOW_AUTOSIZE);
cv::moveWindow(advertise_window_name, 1000, 50);
//find screen resolution

GetDesktopResolution(horizontal, vertical);

//read *.png images from folder "images"
vector<cv::String> fn, fn_depth;
cv::Mat image_input, image_depth, frame_last;

//vector<Mat> images, images_depth;
int image_number = 0;
int frame_nr = 1;
glob("images/*.png", fn, false);
size_t count = fn.size(); //number of png files in images
folder

glob("images_depth/*.png", fn_depth, false);
      size_t count2 = fn_depth.size(); //number of png files
in images_depth folder      if (count2 != count) {
```

```
            cout << "Number of depth maps not equal with the
number of images. Exiting." << endl;
            return 0;
      }

      image_input = imread(fn[image_number], IMREAD_COLOR);
      image_depth       =       imread(fn_depth[image_number],
IMREAD_GRAYSCALE);
}
```

Učitavanje izvornih slika i dubinskih mapa

```
int c = cv::waitKey(10);
if ((char)c == 'c')
{
      break;
      }

if ((char)c == 'f') {
      imwrite("frame_warped_image.png", edited);
      imwrite("frame_mask.png", 255 * mask);
      imwrite("frame_occlusions.png", result2);
            }
if ((char)c == 'n') {
      if (image_number + 1 < count) {
      image_number = image_number + 1;
      image_input = imread(fn[image_number], IMREAD_COLOR);
      image_depth = imread(fn_depth[image_number],
IMREAD_GRAYSCALE);
}
return 0;
}
```

Definiranje komandi (tipke "F"," N" i" C")

```
while (1) {
            frame_nr++;

            time_point<high_resolution_clock> start =
high_resolution_clock::now();
            detectAndDisplay(fn, fn_depth, image_input,
image_depth, horizontal, vertical);

            time_point<high_resolution_clock> stop =
high_resolution_clock::now();
            cout << duration<double, std::milli>(stop -
start).count() << " ms" << endl;

void detectAndDisplay(vector<String> fn, vector<String>
fn_depth, cv::Mat image_input, cv::Mat image_depth, int
horizontal, int vertical) {
      //std::vector<cv::Rect> faces, faces2;
```

```
	//cv::Mat frame_gray;

	vector<Point2f> corners;
	double gamma = 0; double phi = 0;

	double scaleX = 128; double scaleY = 128;
	cv::Rect maxRect; // 0 sized rect
	cv::Vec3b* currentRow, * currentRow_input; uchar*
currentRow_depth;
	int different = 1;
	int pom_x, pom_y;
	POINT point;


	//-- Show what you got

	if (GetCursorPos(&point))
	{

		if (check_x == -1 && check_y == -1) { //only in
first frame



			check_x = point.x;
			check_y = point.y;

			different = 1;
		}
		else {

			pom_x = point.x;
			pom_y = point.y;

			if (abs(pom_x - check_x) <= 7 && abs(pom_y -
check_y) <= 2) { //don't calculate new projection if new
center is less than x pixels from old
				different = 0;
			}
			else {
				different = 1;
				check_x = pom_x;
				check_y = pom_y;
			}

		}

		if (different == 1) {
			//calculate theta and phi from maxRect
			gamma = -(((float)point.x - horizontal / 2) /
(horizontal)) * constant_x - 90;
			phi = -(((float)point.y - vertical / 2) /
(vertical)) * constant_y;
```

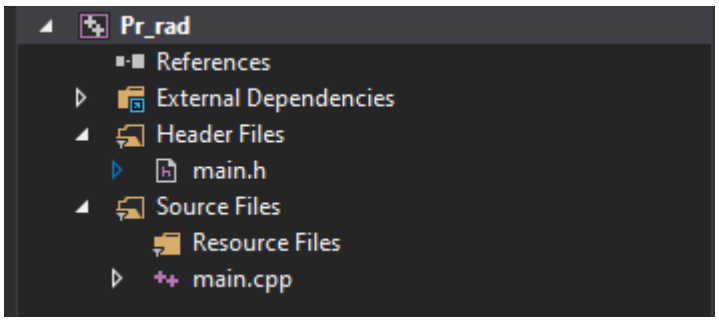
```
                cout << point.x << endl;
                cout << point.y << endl;


                edited = Project3D(image_input, image_depth,
gamma, phi);

                cv::imshow(advertise_window_name, edited);
            }
      }



}

Mat Project3D(Mat& image, Mat& depthmap, double& angleX,
double& angleY)
{

      Mat result = Mat::zeros(image.rows, image.cols,
CV_8UC3);

      int x, y, d;

      double camera_x = cos(3.14159 * 2.0 * angleX / 360) * h;
// camera position
      double camera_y = sin(3.14159 * 2.0 * angleY / 360) * h;



      Mat current_depth(image.rows, image.cols, CV_8UC1);
      current_depth = 0;

      //some math calculate outside for loops
      double atan_x = atan(camera_x / D);
      double cos_atan_x = cos(atan_x);
      double atan_y = atan(camera_y / D);
      double cos_atan_y = cos(atan_y);

      //double
atan_x_approx,tan_x_approx,atan_y_approx,tan_y_approx,x_help,x
_help2,y_help,y_help2;
      double x_help, y_help;

      double pixel_help;

      for (int i = 0; i < image.rows; i++) {
            Vec3b* pixel = image.ptr<Vec3b>(i);
            uchar* depth = depthmap.ptr<uchar>(i);

            for (int j = 0; j < image.cols; j++) {
```

```
            d = max((int)depth[j], 1); // depth of current
pixel, minimum is 1 so that 0 can be used for later mask

x_help = ((camera_x * (L / D + 1) + image.cols / 2 - j) / (D -
d + L))

x = image.cols / 2 - (int)(L * (x_help - atan_x) / ((1 +
x_help * atan_x) * cos_atan_x));

y_help = ((camera_y * (L / D + 1) + image.rows / 2 - i) / (D -
d + L));

y = image.rows / 2 - (int)(L * (y_help - atan_y) / ((1 +
y_help * atan_y) * cos_atan_y));


if (x >= 0 && x < image.cols && y >= 0 && y < image.rows && d
> current_depth.at<uchar>(y, x)) {


result.at<Vec3b>(y, x) = Vec3b(pixel[j][0], pixel[j][1],
pixel[j][2]);
      current_depth.at<uchar>(y, x) = d;

}

}
}

      cv::Mat convert, mask_help, mask_help2;

//create mask for inpainting method later - check for 0 pixels
in current_depth
      threshold(current_depth, mask_help, 0, 1,
THRESH_BINARY_INV);

      //now cut off edges
result.copyTo(result2);

convert = result(Rect(LEFT * result.cols, TOP * result.rows,
(1 - 2 * LEFT) * result.cols, (1 - 2 * TOP) * result.rows));

mask = mask_help(Rect(LEFT * mask_help.cols, TOP *
mask_help.rows, (1 - 2 * LEFT) * mask_help.cols, (1 - 2 * TOP)
* mask_help.rows));


Mat distance, labels;   //destination array

      distanceTransform(mask, distance, labels, DIST_L2,
DIST_MASK_PRECISE, DIST_LABEL_PIXEL);

      //map labels to indices
```

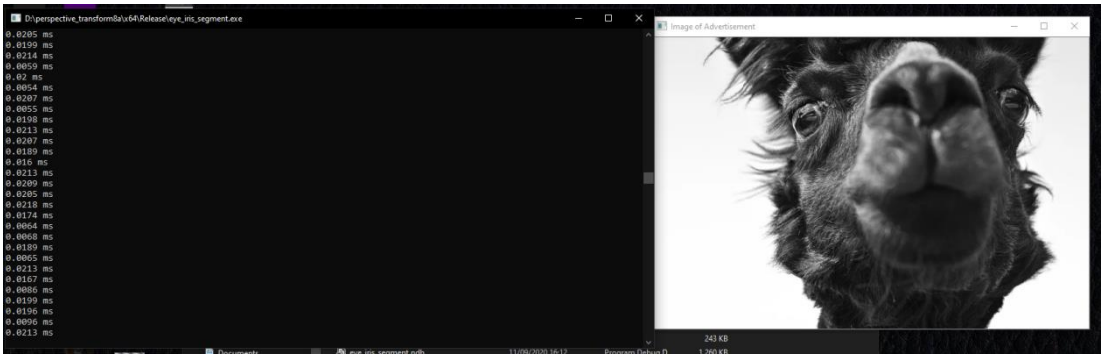
```
	std::vector<cv::Vec2i> label_to_index;

	//reserve memory for faster push_back
	label_to_index.reserve(sum(~mask)[0]);



	//label_to_index.push_back(-1);
	for (int row = 0; row < mask.rows; ++row)
		for (int col = 0; col < mask.cols; ++col)
			if (mask.at<uchar>(row, col) == 0) //this
pixel exist e.g. it is original
				label_to_index.push_back(cv::Vec2i(row,
col));


	for (int row = 0; row < mask.rows; ++row) {
		for (int col = 0; col < mask.cols; ++col) {
			if (mask.at<uchar>(row, col) > 0) {

convert.at<Vec3b>(row, col) =
Vec3b(convert.at<Vec3b>(label_to_index[labels.at<int>(row,
col)])[0],

convert.at<Vec3b>(label_to_index[labels.at<int>(row,
col)])[1],
convert.at<Vec3b>(label_to_index[labels.at<int>(row,
col)])[2]);
			}
		}
	}


	//blur image, but only part with dilated mask

	Mat convert_blurred, mask2;
	GaussianBlur(convert, convert_blurred, Size(5, 5), 0,
0);

	//dilate mask
	mask.copyTo(mask2);
	dilate(mask, mask2, element);


	//create final image as combination from blurred and
original images
	Mat help1, help2;
	bitwise_and(convert_blurred, convert_blurred, help1,
mask2);
	bitwise_and(convert, convert, help2, 1 - mask2);

	add(help1, help2, convert);
```

```
	return convert;
}

// Get the horizontal and vertical screen sizes in pixel
void GetDesktopResolution(int& horizontal, int& vertical)
{
	RECT desktop;
	// Get a handle to the desktop window
	const HWND hDesktop = GetDesktopWindow();
	// Get the size of screen to the variable desktop
	GetWindowRect(hDesktop, &desktop);
	// The top left corner will have coordinates (0,0)
	// and the bottom right corner will have coordinates
	// (horizontal, vertical)
	horizontal = desktop.right;
	vertical = desktop.bottom;
}
```

Izračun koordinata, slika, izobličavanje slike pomakom miša, stvaranje maske i popravljene slike Gaussian algoritmom.



Slika 6.1 - Sadržaj programa.



Slika 6.2 - Prikaz rada programa.

Slika 6.3 - Prikaz originalnih korištenih slika i njihovih dubinskih mapa.

6.2. Prikaz popravljenih slika pomoću tri algoritma

Ovo poglavlje prikazuje gore navedenih pet slika iz različitih pogleda koji dovode do nastanka disokluzija. U poglavlju su slike izobličene, prikazane su njihove maske te krajnji popravljeni izgled, za svaki algoritam posebno.

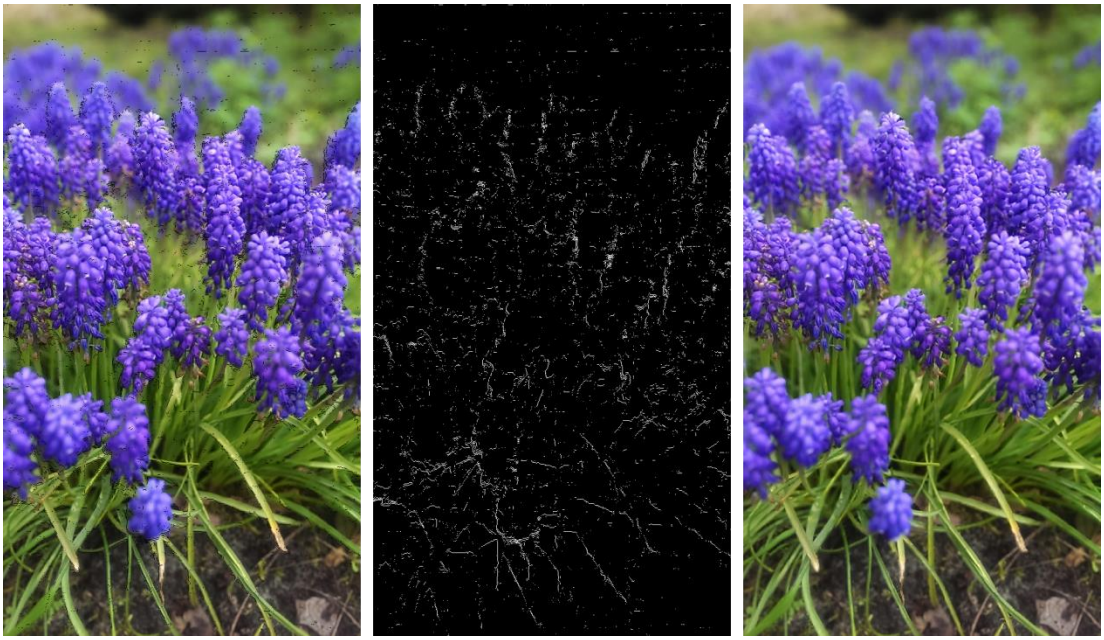
6.2.1. Distance transform + Gaussian Filter

Poglavlje prikazuje pet pogleda – miš desno, lijevo, gore, dolje, desni gornji i lijevi donji kut. Isti pogledi koristit će se i kod drugih algoritama radi točnije usporedbe.

Redoslijed je: Prikaz slika s disokluzija, maska te popravljena slika.

Slika 6.4 - Prikaz prve slike s pogledom u desno (Gaussian).

Slika 6.5 - Prikaz prve slike s pogledom u lijevo (Gaussian).



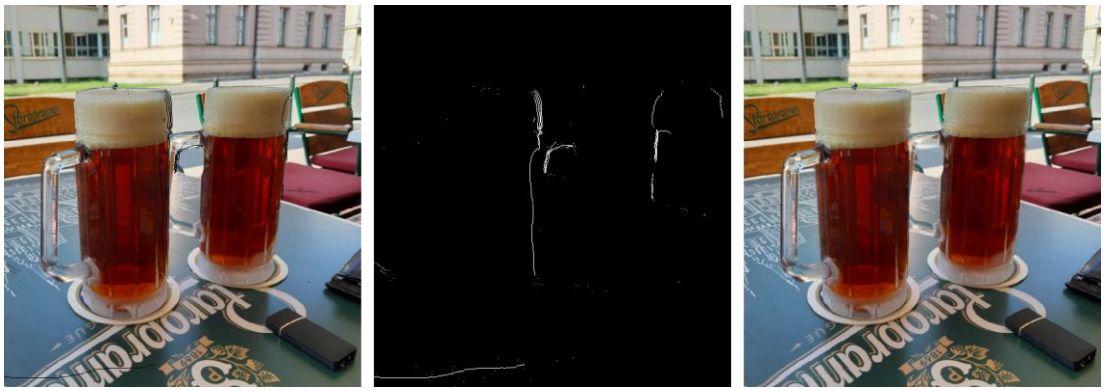
Slika 6.6 - Prikaz prve slike s pogledom prema gore (Gaussian).

Slika 6.7 - Prikaz prve slike s pogledom prema dolje (Gaussian).

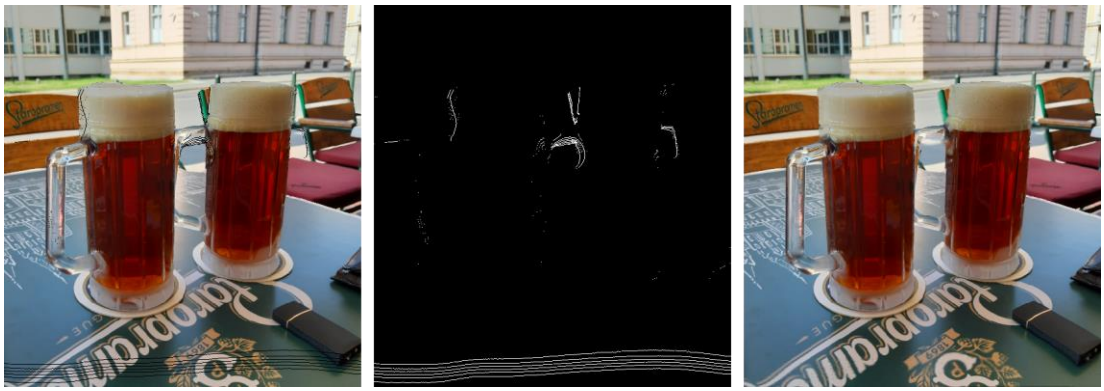
Slika 6.8 - Prikaz prve slike s pogledom u gornji desni kut (Gaussian).

Slika 6.9 - Prikaz prve slike s pogledom u donji lijevi kut (Gaussian).

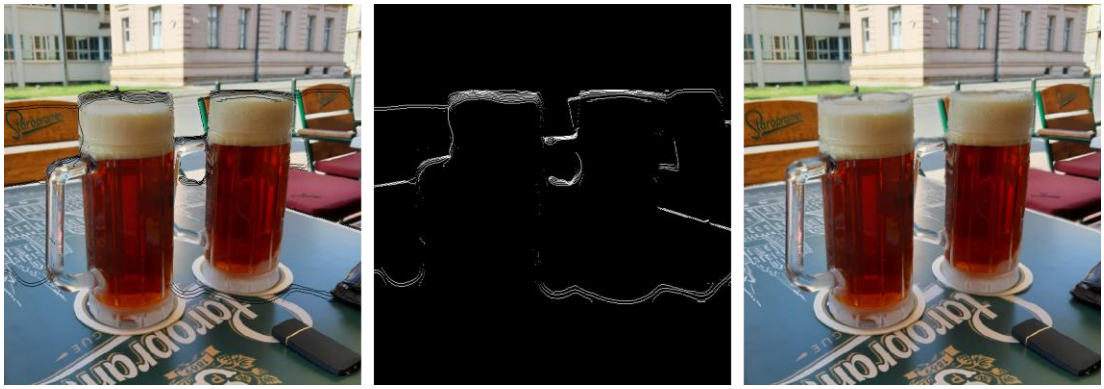
Slika 6.10 - Prikaz druge slike s pogleodm u desno (Gaussian).



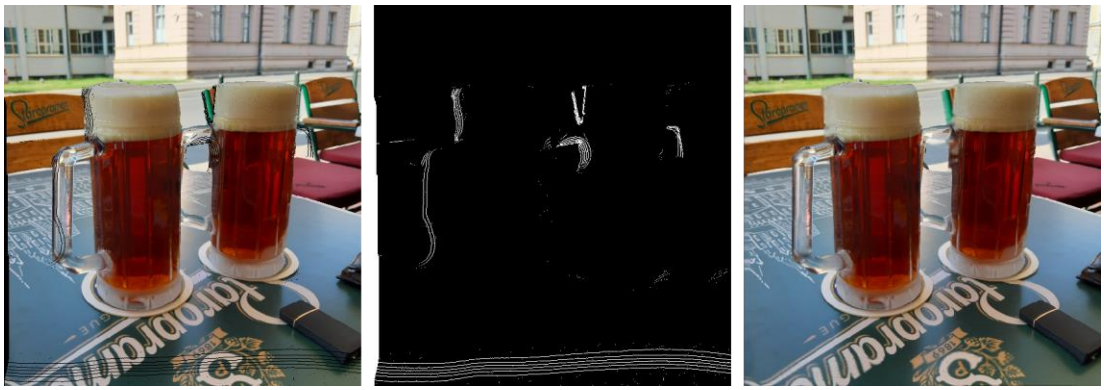
Slika 6.11 - Prikaz druge slike s pogledom u lijevo (Gaussian).



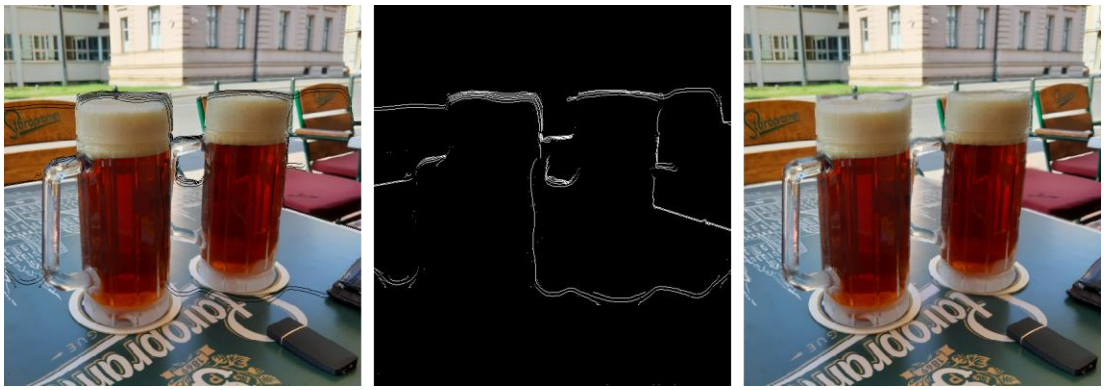
Slika 6.12 - Prikaz druge slike s pogledom prema gore (Gaussian).



Slika 6.13 - Prikaz druge slike s pogledom prema dolje (Gaussian).



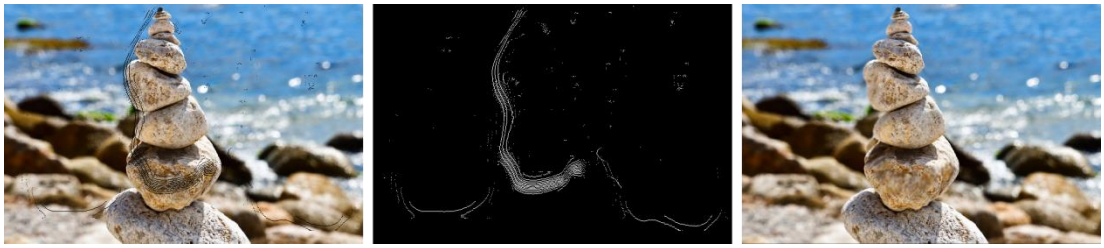
Slika 6.14 - Prikaz druge slike s pogledom u gornji desni kut (Gaussian).



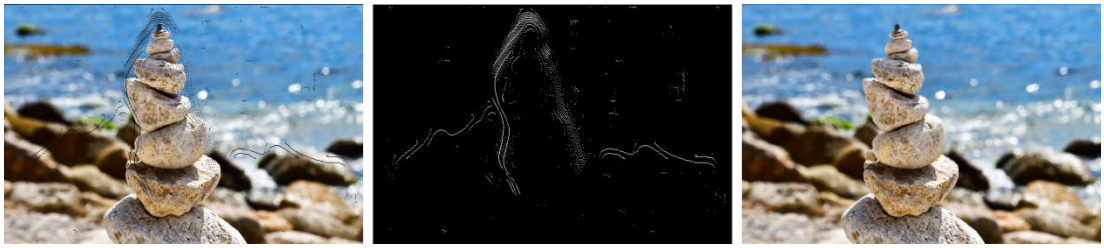
Slika 6.15 - Prikaz druge slike s pogledom u donji lijevi kut (Gaussian).

Slika 6.16 - Prikaz treće slike s pogledom u desno (Gaussian).

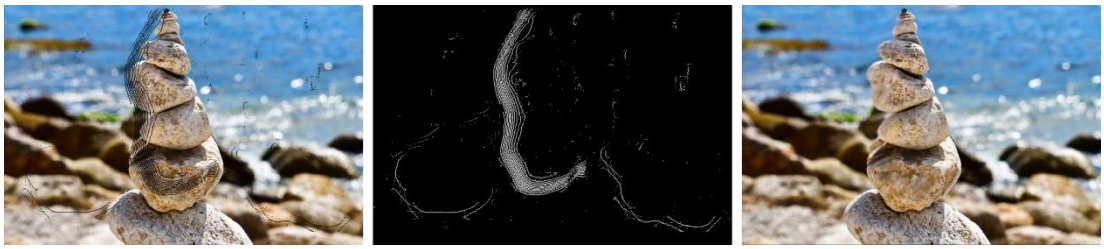
Slika 6.17 - Prikaz treće slike s pogledom u lijevo (Gaussian).



Slika 6.18 - Prikaz treće slike s pogledom prema gore (Gaussian).

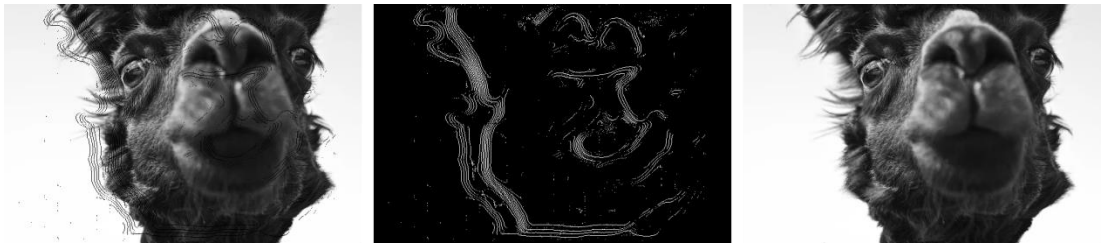


Slika 6.19 - Prikaz treće slike s pogledom prema dolje (Gaussian).

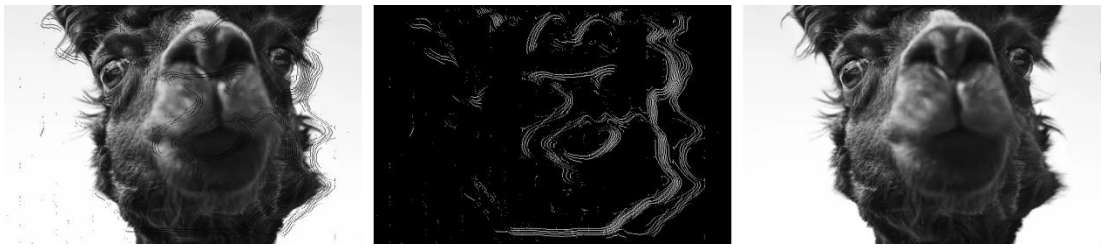


Slika 6.20 - Prikaz treće slike s pogledom u gornji desni kut (Gaussian).

Slika 6.21 - Prikaz treće slike s pogledom u donji lijevi kut (Gaussian).



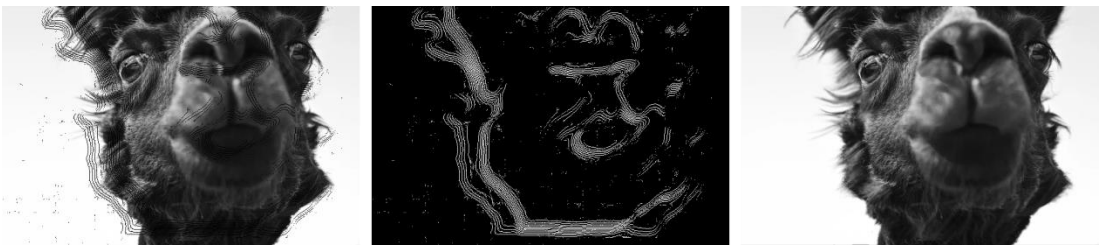
Slika 6.22 - Prikaz četvrte slike s pogledom u desno (Gaussian).



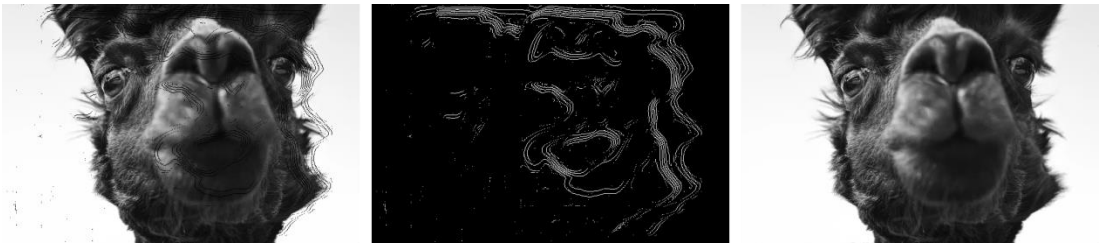
Slika 6.23 - Prikaz četvrte slike s pogledom u lijevo (Gaussian).

Slika 6.24 - Prikaz četvrte slike s pogledom prema gore (Gaussian).

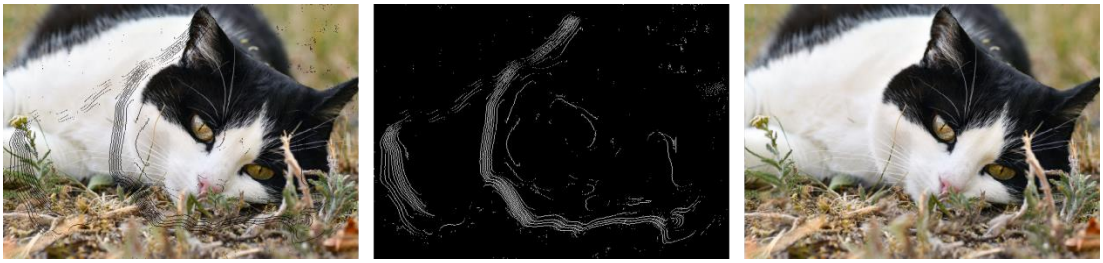
Slika 6.25 - Prikaz četvrte slike s pogledom prema dolje (Gaussian).



Slika 6.26 - Prikaz četvrte slike s pogledom u gornji desni kut (Gaussian).

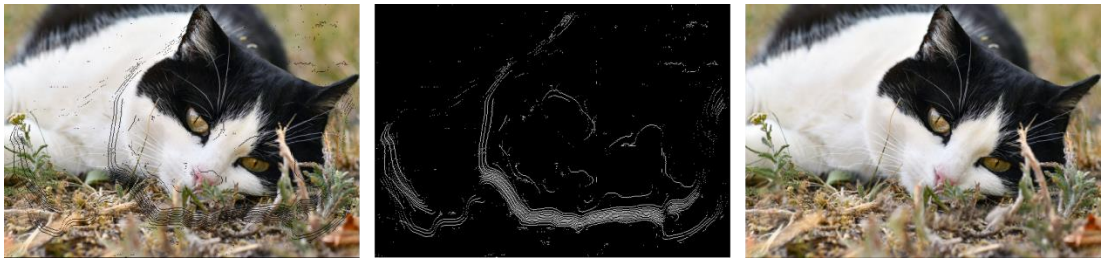


Slika 6.27 - Prikaz četvrte slike s pogleodm u donji lijevi kut (Gaussian).

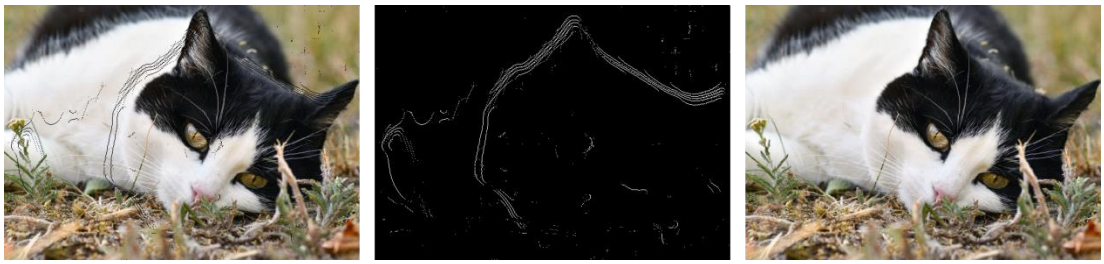


Slika 6.28 - Prikaz pete slike s pogledom u desno (Gaussian).

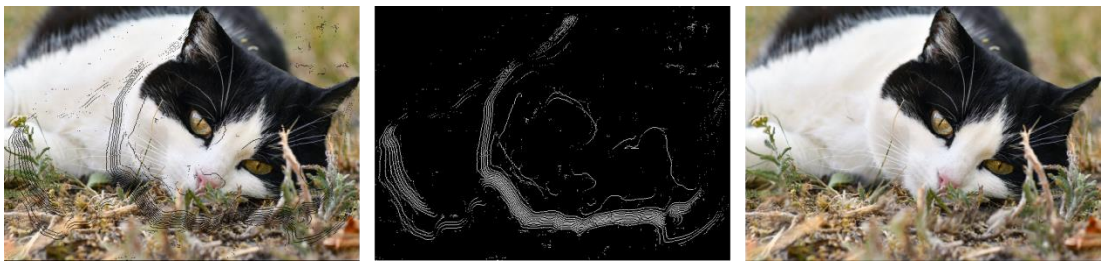
Slika 6.29 - Prikaz pete slike s pogledom u lijevo (Gaussian).



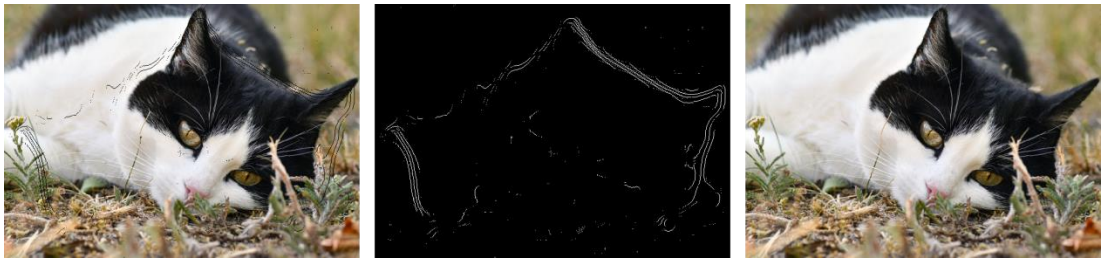
Slika 6.30 - Prikaz pete slike s pogledom prema gore (Gaussian).



Slika 6.31 - Prikaz pete slike s pogledom prema dolje (Gaussian).



Slika 6.32 - Prikaz pete slike s pogledom u gornji desni kut (Gaussian).



Slika 6.33 - Prikaz pete slike s pogledom u donji lijevi kut (Gaussian).

6.2.2. Criminisijev algoritam

Poput prijašnjeg poglavlja, prikazuju se slike popravljene drugim algoritmom.
Redoslijed i pogledi ostaju isti.

```
>> I = imread('d3.png');
>> mask = imread('d3m.png');
>> mask = logical(rgb2gray(mask));
>> J =
inpaintExemplar(I,mask,'FillOrder','tensor','PatchSize',7);
>> montage({J});
```

Slika 6.34 - Prikaz prve slike s pogledom u desno (Criminisi).

Slika 6.35 - Prikaz prve slike s pogledom u lijevo (Criminisi).

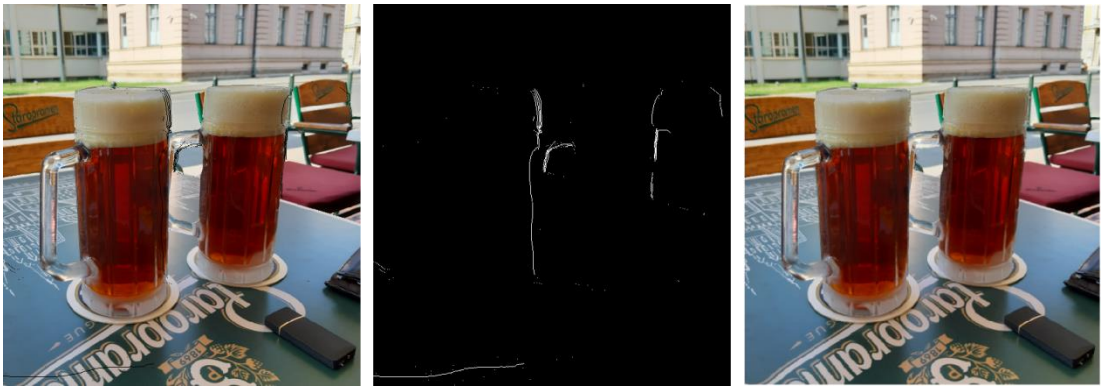
Slika 6.36 - Prikaz prve slike s pogledom prema gore (Criminisi).

Slika 6.37 - Prikaz prve slike s pogledom prema dolje (Criminisi).

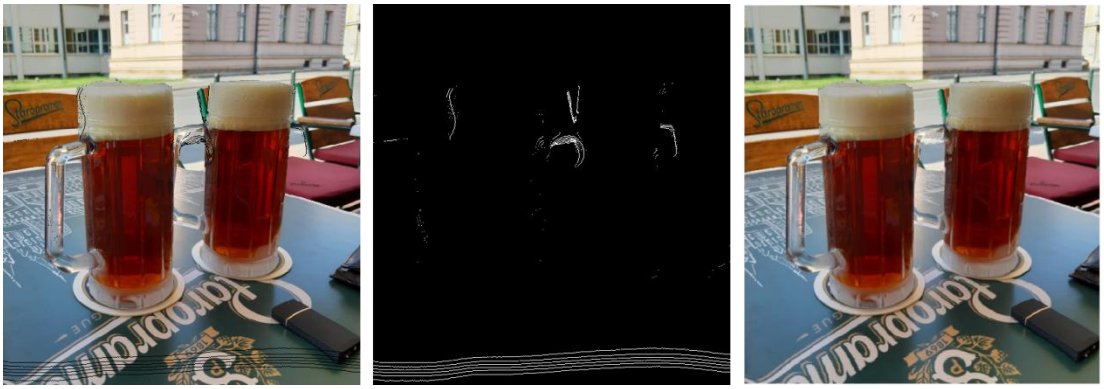
Slika 6.38 - Prikaz prve slike s pogledom u gornji desni kut (Criminisi).

Slika 6.39 - Prikaz prve slike s pogledom u donji lijevi kut (Criminisi).

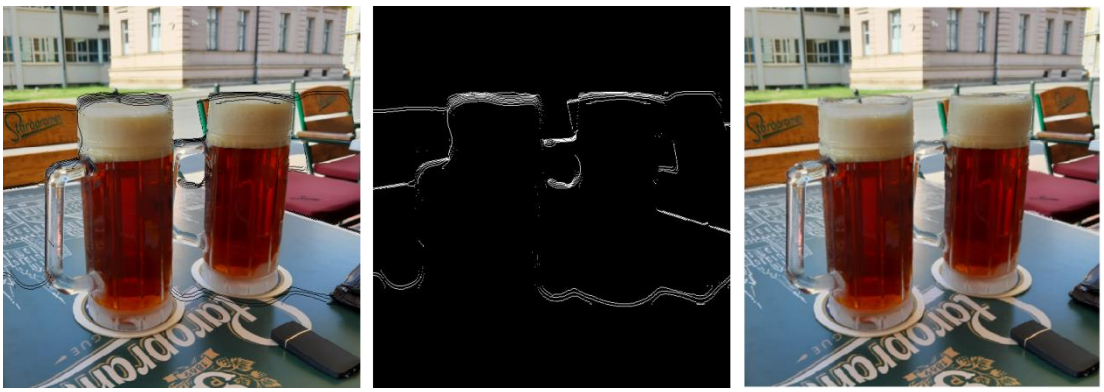
Slika 6.40 - Prikaz druge slike s pogledom u desno (Criminisi).



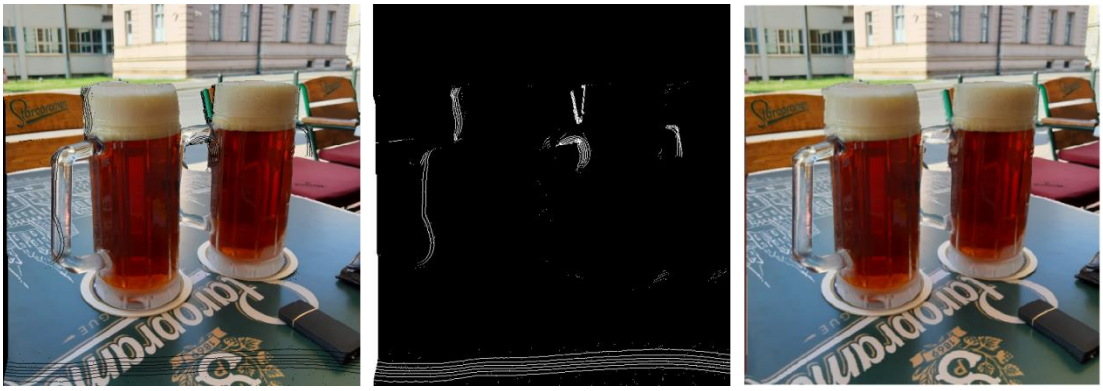
Slika 6.41 - Prikaz druge slike s pogledom u lijevo (Criminisi).



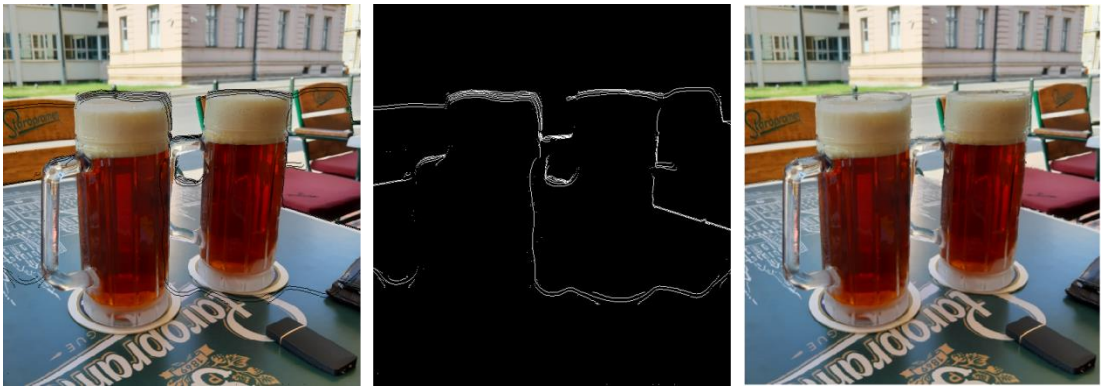
Slika 6.42 - Prikaz druge slike s pogledom prema gore (Criminisi).



Slika 6.43 - Prikaz druge slike s pogleodm prema dolje (Criminisi).



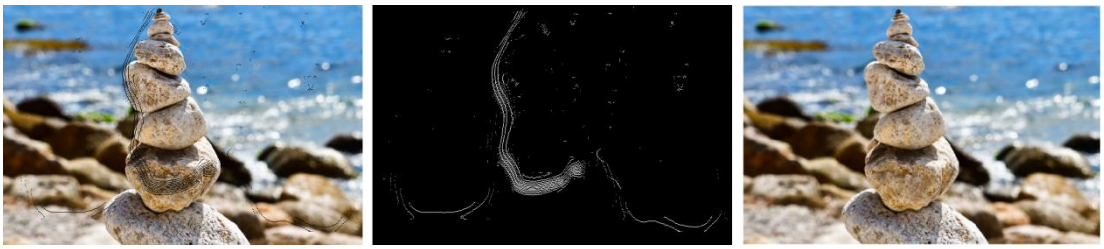
Slika 6.44 - Prikaz druge slike s pogledom u gornji desni kut (Criminisi).



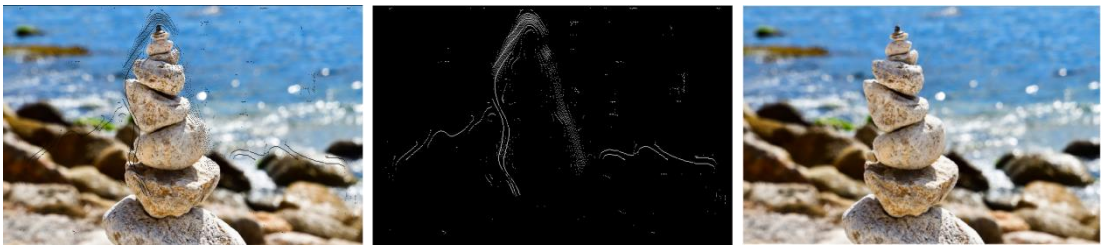
Slika 6.45 - Prikaz druge slike s pogledom u donji lijevi kut (Criminisi).

Slika 6.46 - Prikaz treće slike s pogledom u desno (Criminisi).

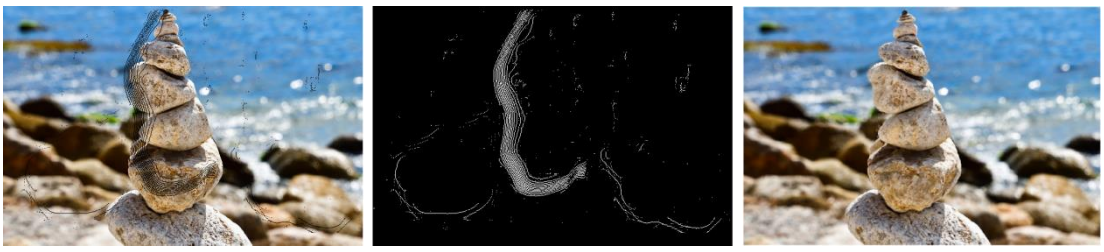
Slika 6.47 - Prikaz treće slike s pogledom u lijevo (Criminisi).



Slika 6.48 - Prikaz treće slike s pogledom prema gore (Criminisi).



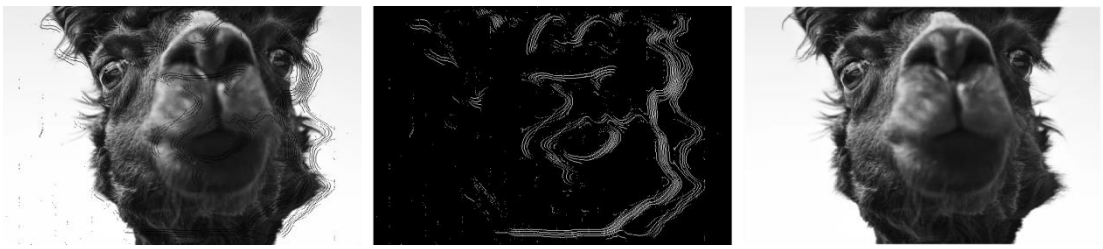
Slika 6.49 - Prikaz treće slike s pogledom prema dolje (Criminisi).



Slika 6.50 - Prikaz treće slike s pogledom u gornji desni kut (Criminisi).

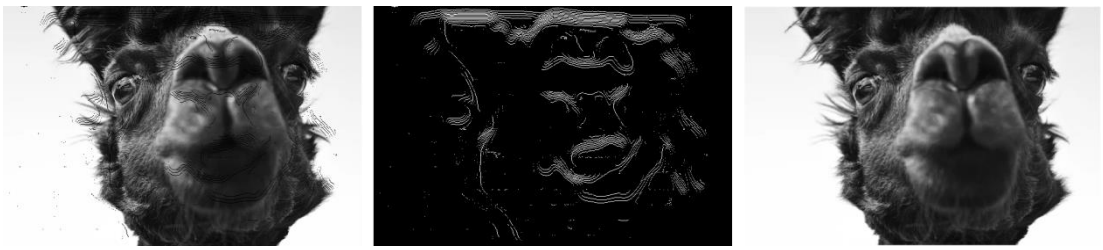
Slika 6.51 - Prikaz treće slike s pogledom u donji lijevi kut (Criminisi).

Slika 6.52 - Prikaz četvrte slike s pogledom u desno (Criminisi).

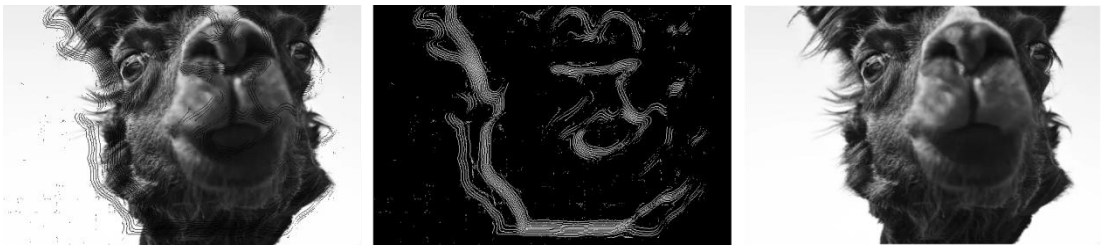


Slika 6.53 - Prikaz četvrte slike s pogledom u lijevo (Criminisi).

Slika 6.54 - Prikaz četvrte slike s pogledom prema gore (Criminisi).

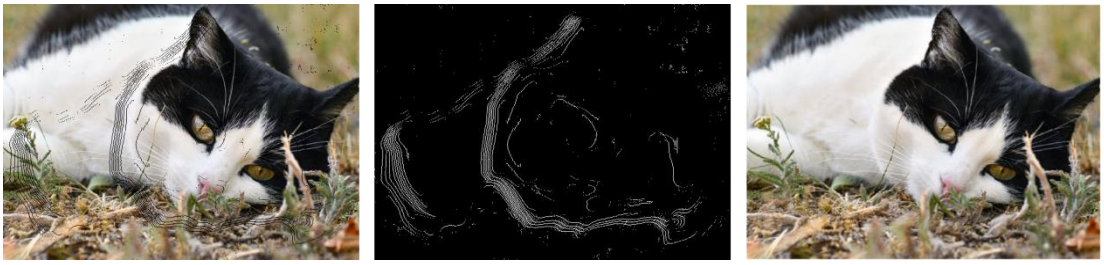


Slika 6.55 - Prikaz četvrte slike s pogledom prema dolje (Criminisi).

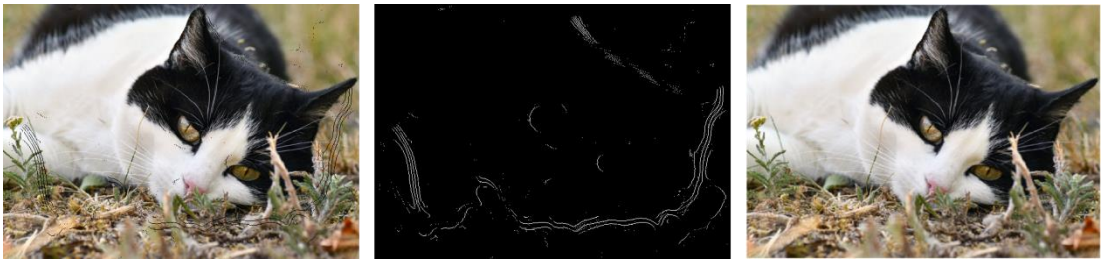


Slika 6.56 - Prikaz četvrte slike s pogledom u gornji desni kut (Criminisi).

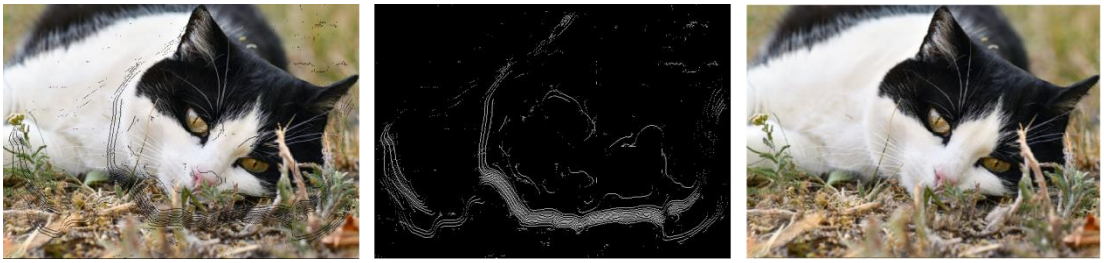
Slika 6.57 - Prikaz četvrte slike s pogledom u donji lijevi kut (Criminisi).



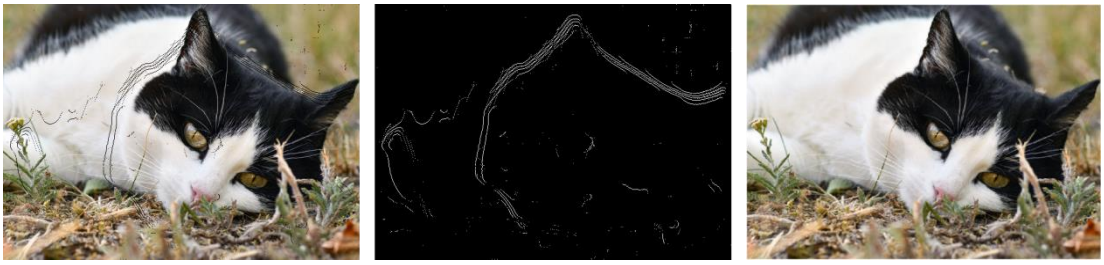
Slika 6.58 - Prikaz pete slike s pogledom u desno (Criminisi).



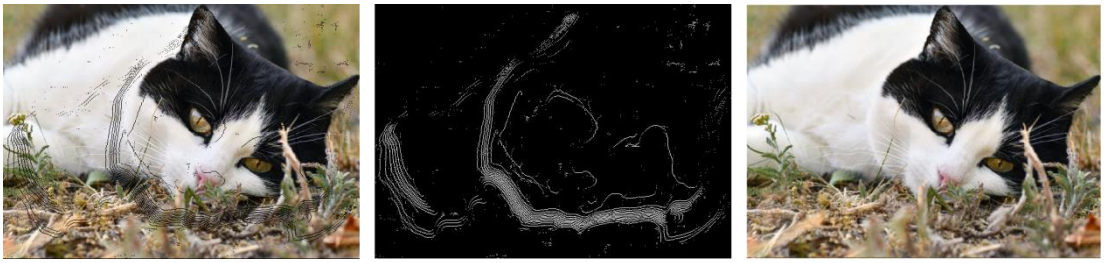
Slika 6.59 - Prikaz pete slike s pogledom u lijevo (Criminisi).



Slika 6.60 - Prikaz pete slike s pogledom prema gore (Criminisi).



Slika 6.61 - Prikaz pete slike s pogledom prema dolje (Criminisi).



Slika 6.62 - Prikaz pete slike s pogledom u gornji desni kut (Criminisi).

Slika 6.63 - Prikaz pete slike s pogledom u donji lijevi kut (Criminisi).

6.2.3. Coherence Transport Based Inpainting

Slikanje temeljeno na koherentnom prijevozu (*eng. Coherence Transport Based Inpainting*), u opisu slika *CTBI*, sljedeći je algoritam pomoću kojeg će se popravljati slike.
Redoslijed i poredak ostaje isti.

```
>> I = imread('cdl5.png');
>> mask = imread('cdl5m.png');
>> mask = logical(rgb2gray(mask));
>> J = inpaintCoherent(I,mask);
>> montage({J});
```

Primjer koda navedenog algoritma

Slika 6.64 - Prikaz prve slike s pogledom u desno (CTBI).

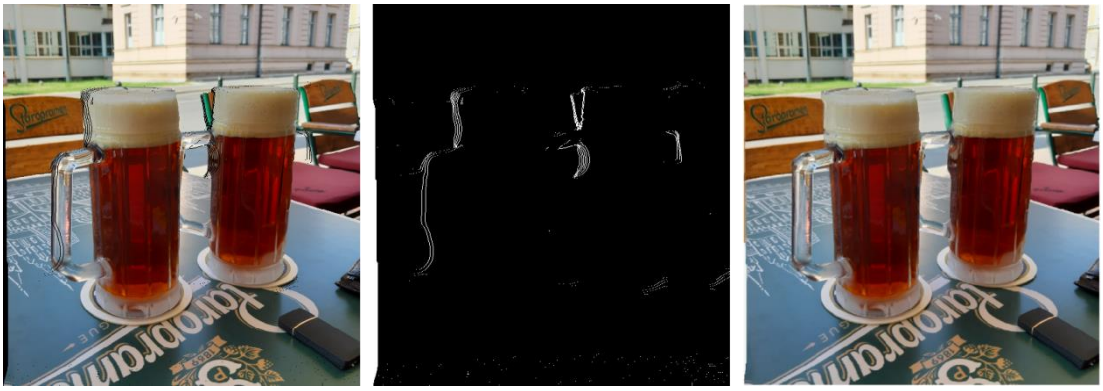
Slika 6.65 - Prikaz prve slike s pogledom u lijevo (CTBI).

Slika 6.66 - Prikaz prve slike s pogledom prema gore (CTBI).

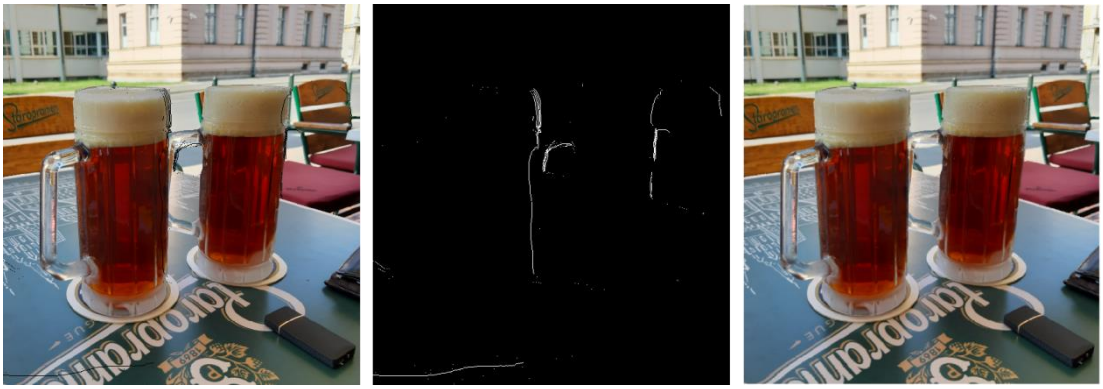
Slika 6.67 - Prikaz prve slike s pogledom prema dolje (CTBI).

Slika 6.68 - Prikaz prve slike s pogledom u gornji desni kut (CTBI).

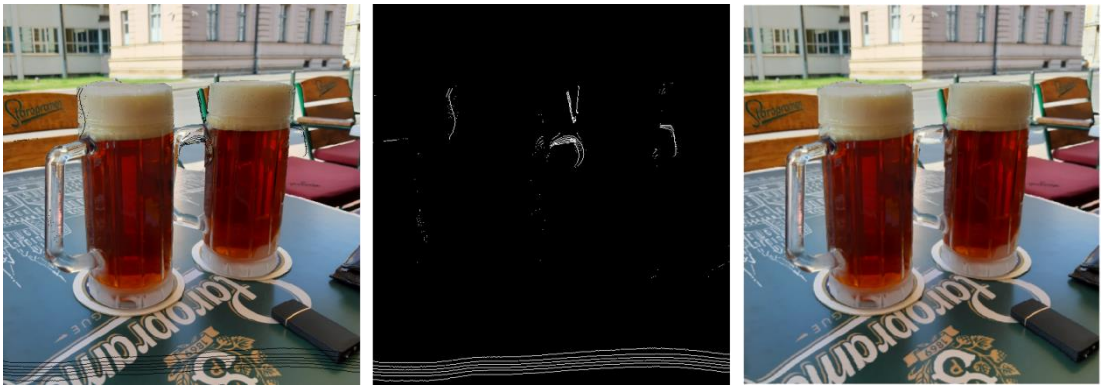
Slika 6.69 - Prikaz prve slike s pogledom u donji lijevi kut (CTBI).



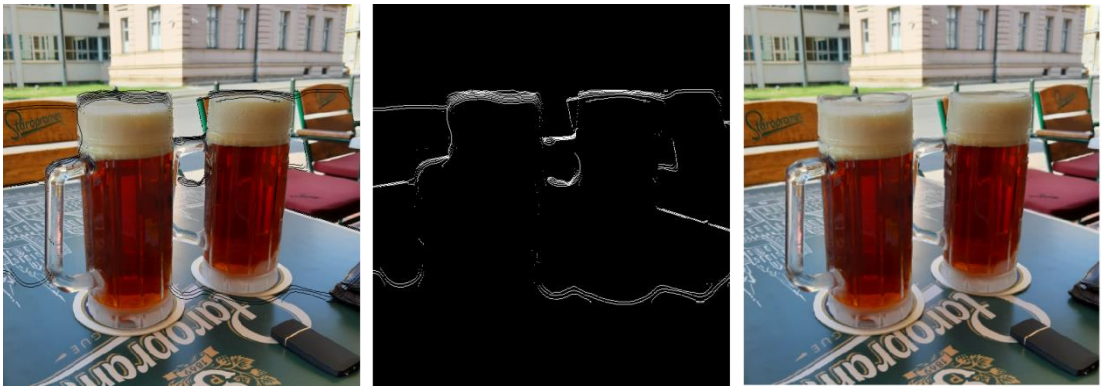
Slika 6.70 - Prikaz druge slike s pogledom u desno (CTBI).



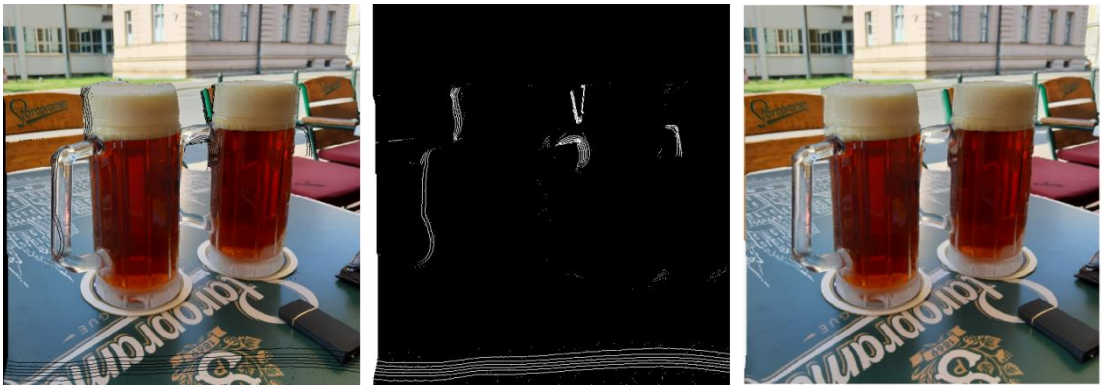
Slika 6.71 - Prikaz druge slike s pogledom u lijevo (CTBI).



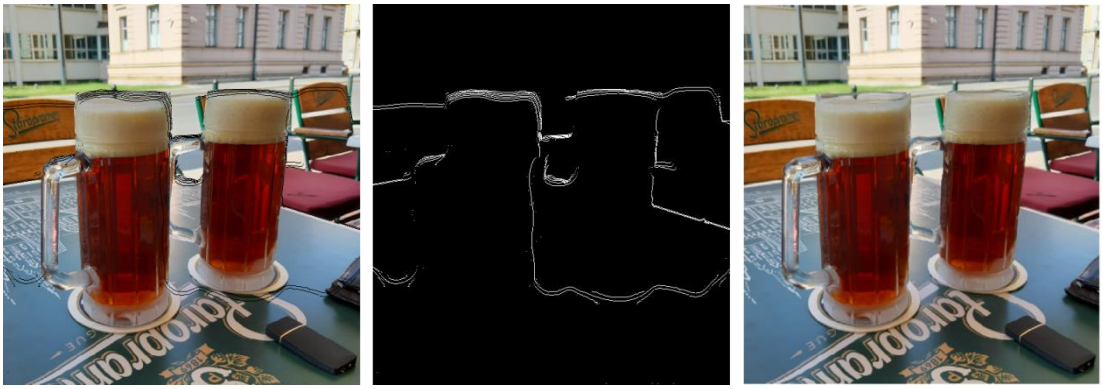
Slika 6.72 - Prikaz druge slike s pogledom prema gore (CTBI).



Slika 6.73 - Prikaz druge slike s pogledom prema dolje (CTBI).



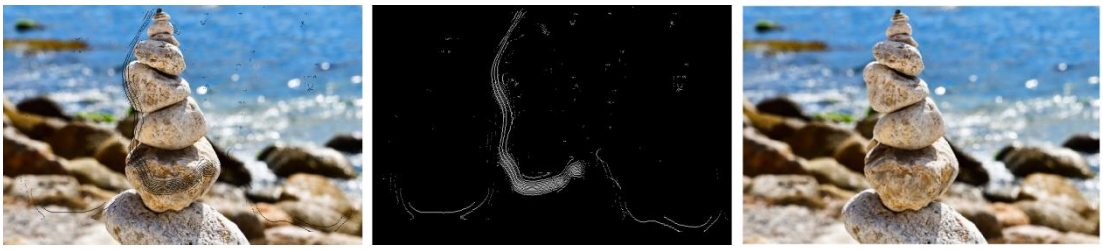
Slika 6.74 - Prikaz druge slike s pogledom u gornji desni kut (CTBI).



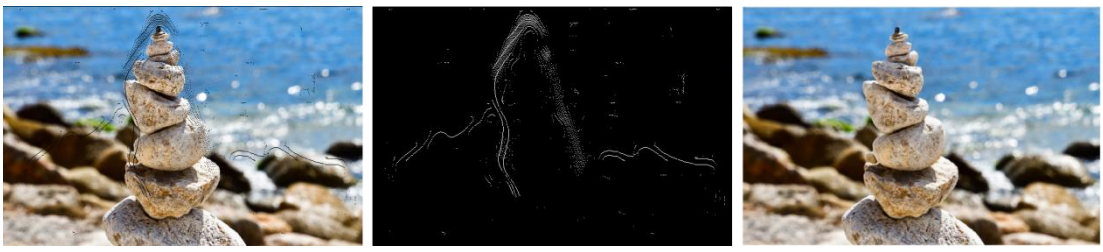
Slika 6.75 - Prikaz druge slike s pogledom u donji lijevi kut (CTBI).

Slika 6.76 - Prikaz treće slike s pogledom u desno (CTBI).

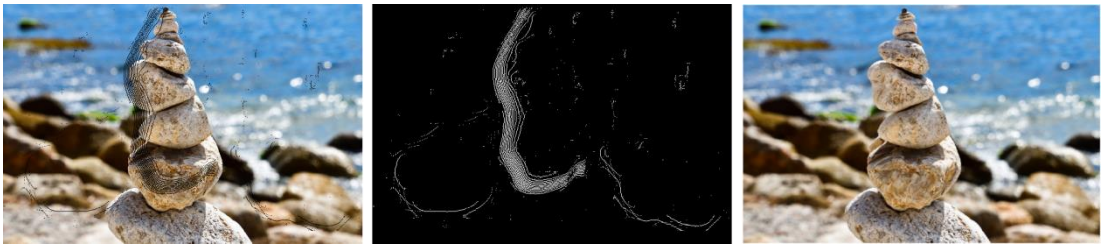
Slika 6.77 - Prikaz treće slike s pogledom u lijevo (CTBI).



Slika 6.78 - Prikaz treće slike s pogledom prema gore (CTBI).



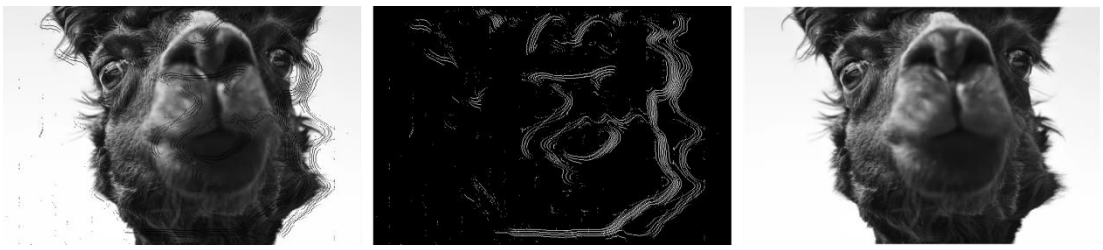
Slika 6.79 - Prikaz treće slike s pogledom prema dolje (CTBI).



Slika 6.80 - Prikaz treće slike s pogledom u gornji desni kut (CTBI).

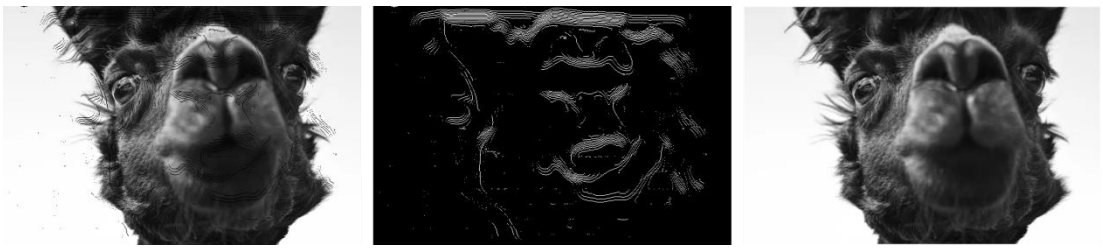
Slika 6.81 - Prikaz treće slike s pogledom u donji lijevi kut (CTBI).

Slika 6.82 - Prikaz četvrte slike s pogledom u desno (CTBI).

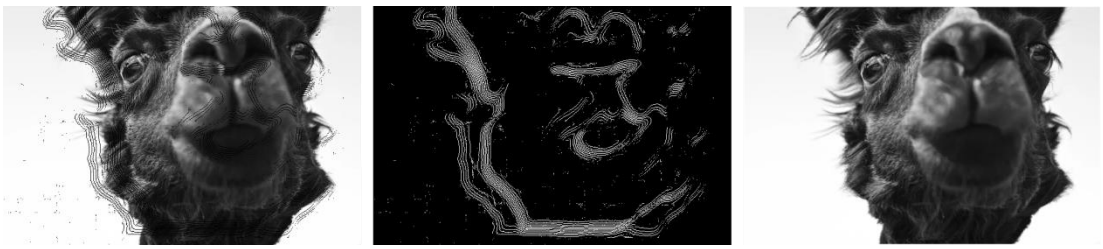


Slika 6.83 - Prikaz četvrte slike s pogledom u lijevo (CTBI).

Slika 6.84 - Prikaz četvrte slike s pogledom prema gore (CTBI).

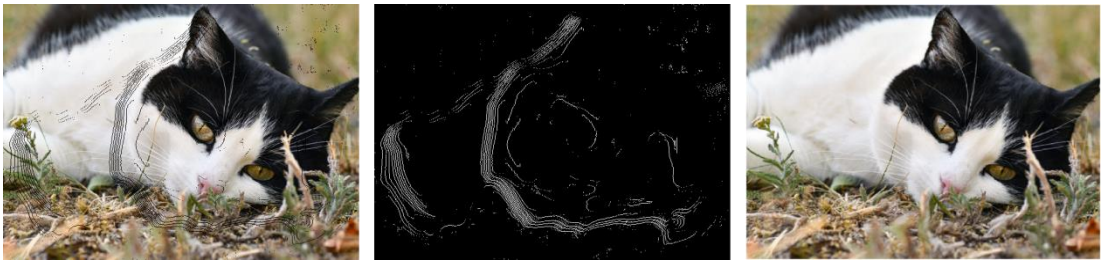


Slika 6.85 - Prikaz četvrte slike s pogledom prema dolje (CTBI).

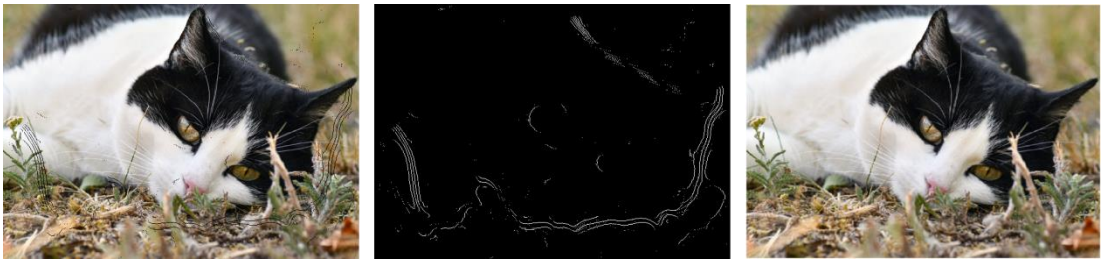


Slika 6.86 - Prikaz četvrte slike s pogledom u gornji desni kut (CTBI).

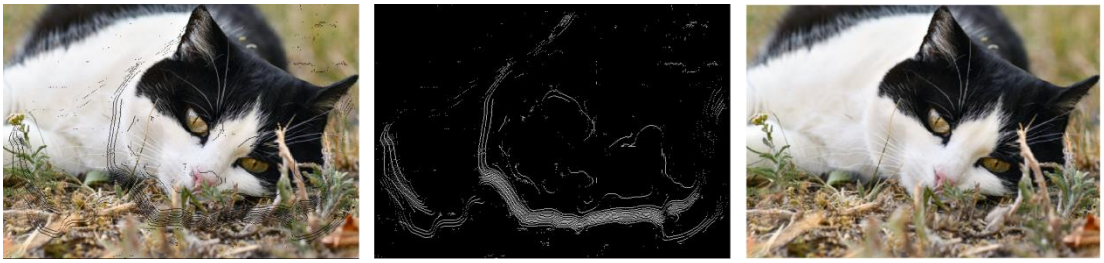
Slika 6.87 - Prikaz četvrte slike s pogledom u donji lijevi kut (CTBI).



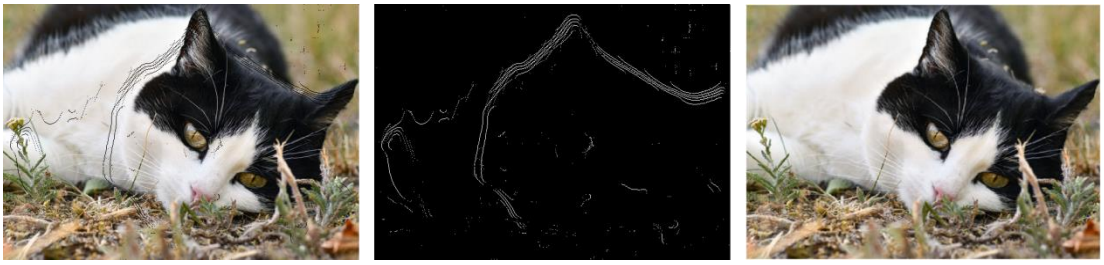
Slika 6.88 - Prikaz pete slike s pogledom u desno (CTBI).



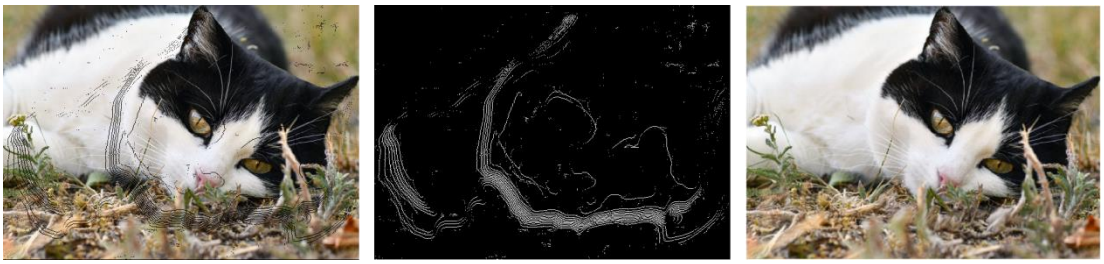
Slika 6.89 - Prikaz pete slike s pogledom u lijevo (CTBI).



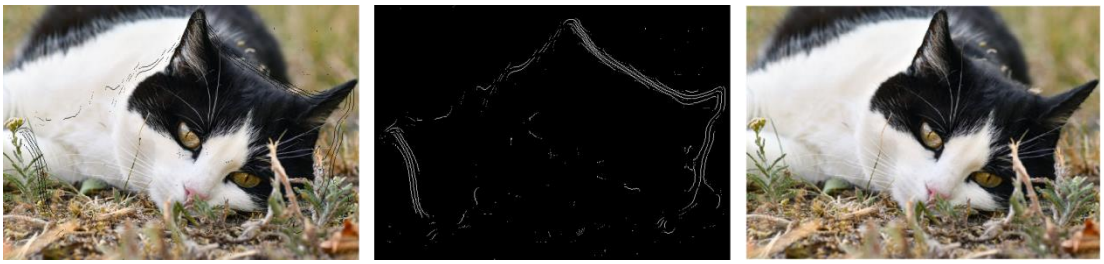
Slika 6.90 - Prikaz pete slike s pogledom prema gore (CTBI).



Slika 6.91 - Prikaz pete slike s pogledom prema dolje (CTBI).



Slika 6.92 - Prikaz pete slike s pogledom u gornji desni kut (CTBI).



Slika 6.93 - Prikaz pete slike s pogleodm u donji lijevi kut (CTBI).

6.3. Objektivno (No-Reference) istraživanje

Prva metoda korištena bodovanje popravljenih slika ova tri algoritma je *BRISQUE* metoda. Slijepi/bez-referentni evaluator prostorne kvalitete slika (*eng. Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE)*) ocjenjuje slike bez korištenja originalne.
Metoda radi na način da se napravi direktorij s izobličenim, popravljenim slikama te se nakon toga jedna po jedna učitavaju i ocjenjuju pomoću unaprijed istreniranih parametara u BRISQUE mjeri.
Ocjene se dodjeljuju za svaki pogled (6 brojeva po slici, 30 ukupno).
Ocjene su u rangu od 0 do 100, a što je broj manji, to su popravljene slike perceptivno bliže početnoj, tj. kvalitetnije su [9].

Ocjene za svaku sliku i algoritam prikazani su tablično.

Slika 1	**Slika 2**	**Slika 3**	**Slika 4**	**Slika 5**
41.3900	56.6324	38.5752	55.4199	57.2863
39.5639	55.6756	38.5256	55.1984	54.7744
41.7330	55.8103	39.4177	55.9602	56.8200
38.4359	56.9270	38.9079	56.0597	55.5406
40.9659	56.2680	38.9490	55.8103	57.3742
37.4339	56.2319	38.9498	55.4131	55.0573

Tablica 1 - BRISQUE metoda - transformacija udaljenosti + Gaussian filter

Slika 1	**Slika 2**	**Slika 3**	**Slika 4**	**Slika 5**
69.5456	56.2916	42.1694	55.9247	37.9526
71.2181	56.5497	41.3818	56.2686	37.6044
70.5300	57.2370	41.1586	55.6052	37.5407
70.1463	55.8738	43.2419	56.3866	38.1875
69.3515	57.5599	41.6919	55.4295	37.8475
70.7596	55.2119	43.6734	56.3129	37.6844

Tablica 2 - BRISQUE metoda - Criminisi

Slika 1	Slika 2	Slika 3	Slika 4	Slika 5
69.1681	58.8087	38.0186	79.5388	23.4449
71.0594	58.8160	38.2063	79.6975	22.8325
70.5010	58.8191	38.0571	79.4825	22.2357
70.0242	58.8209	38.8729	79.8675	24.3545
69.6573	58.8598	38.0012	79.3951	22.8509
70.2531	58.4913	39.2055	79.7281	23.4941

Tablica 3 – BRISQUE metoda - CTBI

Druga metoda korištena za ocjenjivanje popravljenih slika je *NIQE* metoda. Evaluator kvalitete prirodnosti slike (*eng. Naturalness Image Quality Evaluator (NIQE*)) također ocjenjuje slike bez referenci. Metoda radi na sličan način *BRISQUE* metode, niže brojke pokazuju da su popravljene slike perceptivno bliže originalnoj, tj. kvalitetnije.

NIQE mjeri udaljenost između značajki izmjerenih iz svake slike (tj. slika za svaki od 6 pogleda) i značajki dobivenih iz baze podataka slika (tj. unaprijed istreniranih parametara), temeljenih na modelu statistike prirodne scene (*eng. Natural Scene Statistics, NSS*). Model je, kao i kod BRISQUE metode, izveden iz statistike prirodne scene [10].

Ocjene za svaku sliku i algoritam prikazani su tablično.

Slika 1	**Slika 2**	**Slika 3**	**Slika 4**	**Slika 5**
3.9590	3.4263	4.2346	3.8580	3.7271
4.0850	3.6691	4.8663	3.7754	3.3257
4.0129	3.7195	4.4013	3.5927	3.6283
4.1097	3.5319	3.9891	3.7975	3.1093
4.2174	3.6783	4.3700	3.7072	3.8612
4.3095	3.5874	5.0000	4.1184	3.1969

Tablica 4 - NIQE metoda - transformacija udaljenosti + Gaussian filter

Slika 1	**Slika 2**	**Slika 3**	**Slika 4**	**Slika 5**
4.3817	3.7891	3.4092	3.2779	2.6314
4.3253	3.7608	3.4011	3.2411	2.6098
4.4547	3.7785	3.1251	3.1602	2.9097
4.3385	3.8669	3.3575	3.2944	2.6352
4.3551	3.9662	3.2536	3.1120	2.8343
4.4694	3.7583	3.4430	3.3257	2.5837

Tablica 5 - NIQE metoda - Criminisi

Slika 1	Slika 2	Slika 3	Slika 4	Slika 5
4.4289	3.2323	3.5712	3.1406	2.8092
4.3838	3.4708	3.4160	3.0904	2.6823
4.5184	3.0368	3.2896	3.1779	3.0093
4.4470	3.3832	3.7587	3.1362	2.7555
4.4299	3.0444	3.3031	3.2135	3.1138
4.4912	3.3142	3.5471	3.0504	2.7320

Tablica 6 - NIQE metoda - CTBI

Treća korištena metoda za bodovanje popravljenih slika je *PIQE*. Evaluator kvalitete temeljen na percepciji (*eng. Perception Based Quality Evaluator*). Za razliku od prijašnja dva evaluatora, PIQE ne koristi istestirane parametre dobivene iz nekog skupa slika, već odmah učitava sliku i boduje je putem percepcije.

Ocjenjivanje je kao i u BRISQUE metodi, od 0 do 100, i isto manji broj predstavlja višu kvalitetu slike[11].

Bodovi za svaku sliku i algoritam prikazani su tablično.

Slika 1	**Slika 2**	**Slika 3**	**Slika 4**	**Slika 5**
62.9819	51.0643	39.1042	44.7070	40.7219
63.1450	48.8005	37.5817	46.5276	38.0722
59.8695	52.2148	34.9741	44.4715	41.0849
60.1088	53.8537	40.2775	46.3078	36.7492
64.1536	52.6162	37.4988	45.7341	42.8773
61.4550	50.5730	39.5367	45.3606	37.1366

Tablica 7 - PIQE metoda - Transformacija udaljenosti + Gaussian filter

Slika 1	**Slika 2**	**Slika 3**	**Slika 4**	**Slika 5**
39.7154	53.6415	40.3134	33.3976	36.5738
36.1268	48.8431	37.5681	35.7664	38.9715
38.5089	51.6808	38.2091	33.3537	37.9026
40.8360	53.8733	37.9934	35.9803	38.6773
38.6060	52.0790	37.6047	33.2272	37.9691
39.2655	51.8281	35.3782	32.9779	39.4474

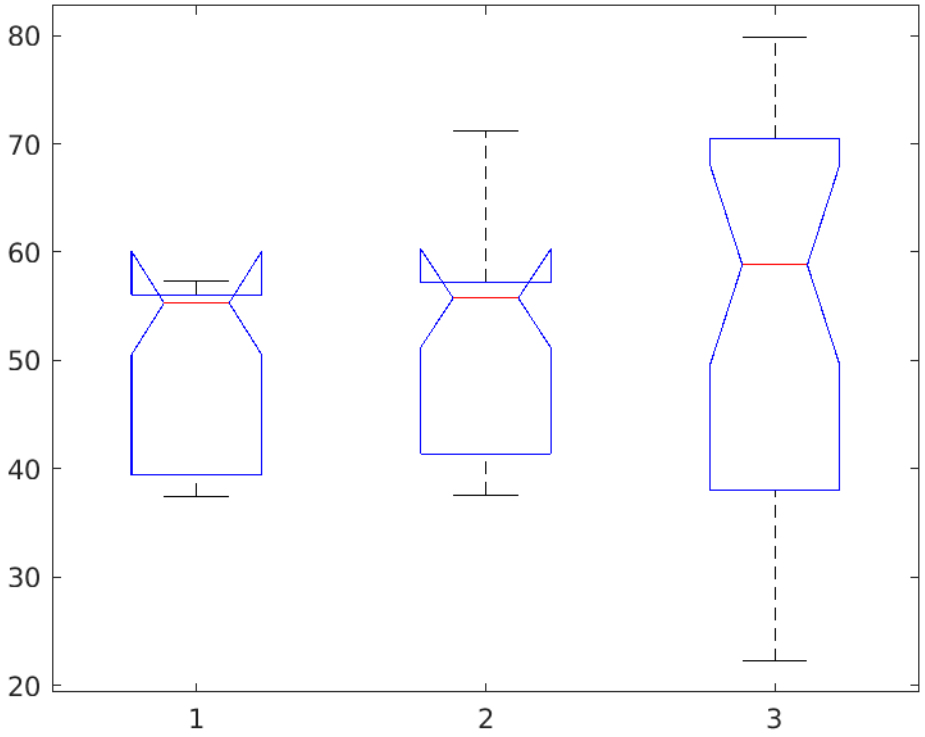
Tablica 8 - PIQE metoda - Criminisi

Slika 1	Slika 2	Slika 3	Slika 4	Slika 5
40.3992	53.3991	42.1688	35.1288	39.1330
38.8472	55.2876	38.5001	34.7274	40.0416
39.6547	56.0278	39.2177	35.3147	37.2277
41.6240	54.9183	39.8087	36.0862	38.9764
39.9015	55.0989	40.0884	35.2492	37.7316
39.5515	55.3691	38.0549	35.3297	36.8202

Tablica 9 - PIQE metoda - CTBI

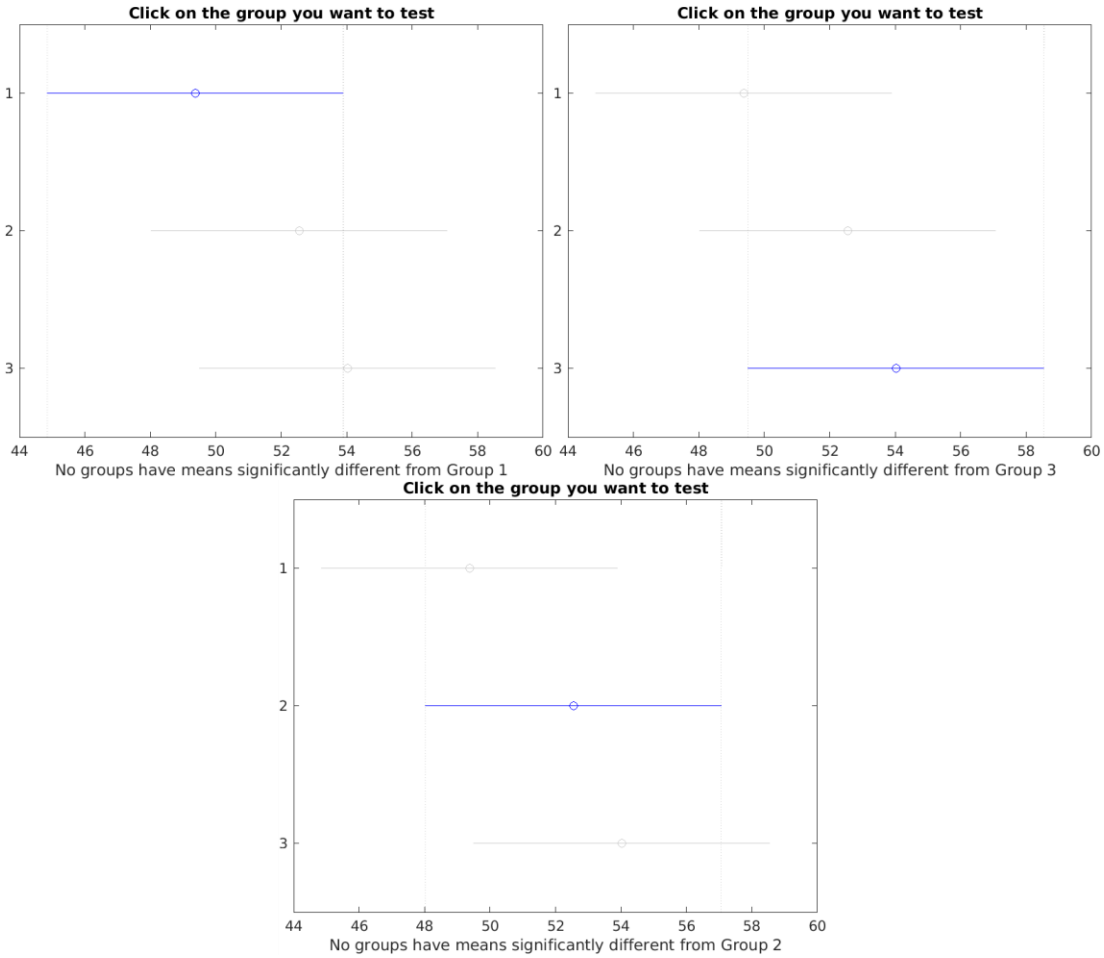
6.3.1. Statistička usporedba ocjena

Usporedba rezultata radi se preko ANOVA i multicompare testa. ANOVA u ovom slučaju ima p vrijednost p = 9.8945e-05, što je manje od 0.05 i znači da neka od mjera ima statistički značajno različitu srednju vrijednost (ili više njih). Multcompare funkcija služi kako bi se zaključilo koja od mjera je različita.



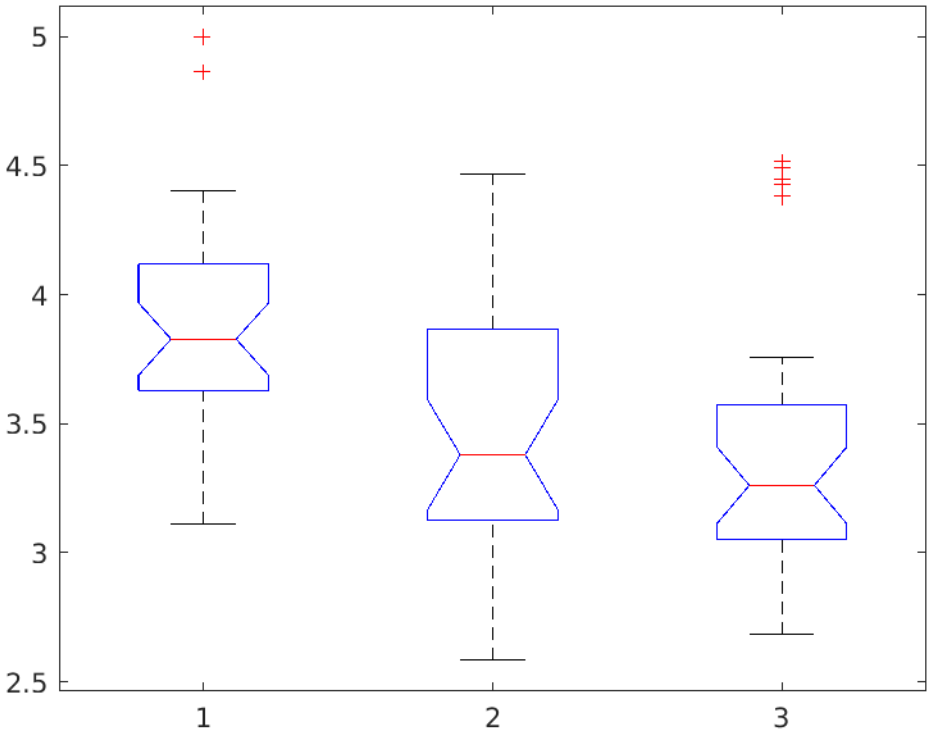
Slika 6.94 - Prikaz ANOVA dijagrama za BRISQUE metodu.

Prvi algoritam (1) – Medijan = 55.3058, Max = 57.3742, Min = 37.4339
Drugi algoritam (2) – Medijan = 55.7395, Max = 71.2181, Min = 37.5407
Treći algoritam (3) – Medijan = 58.8175, Max = 79.8675, Min = 22.2357
F = 0.78, Prob>F = 0.4602



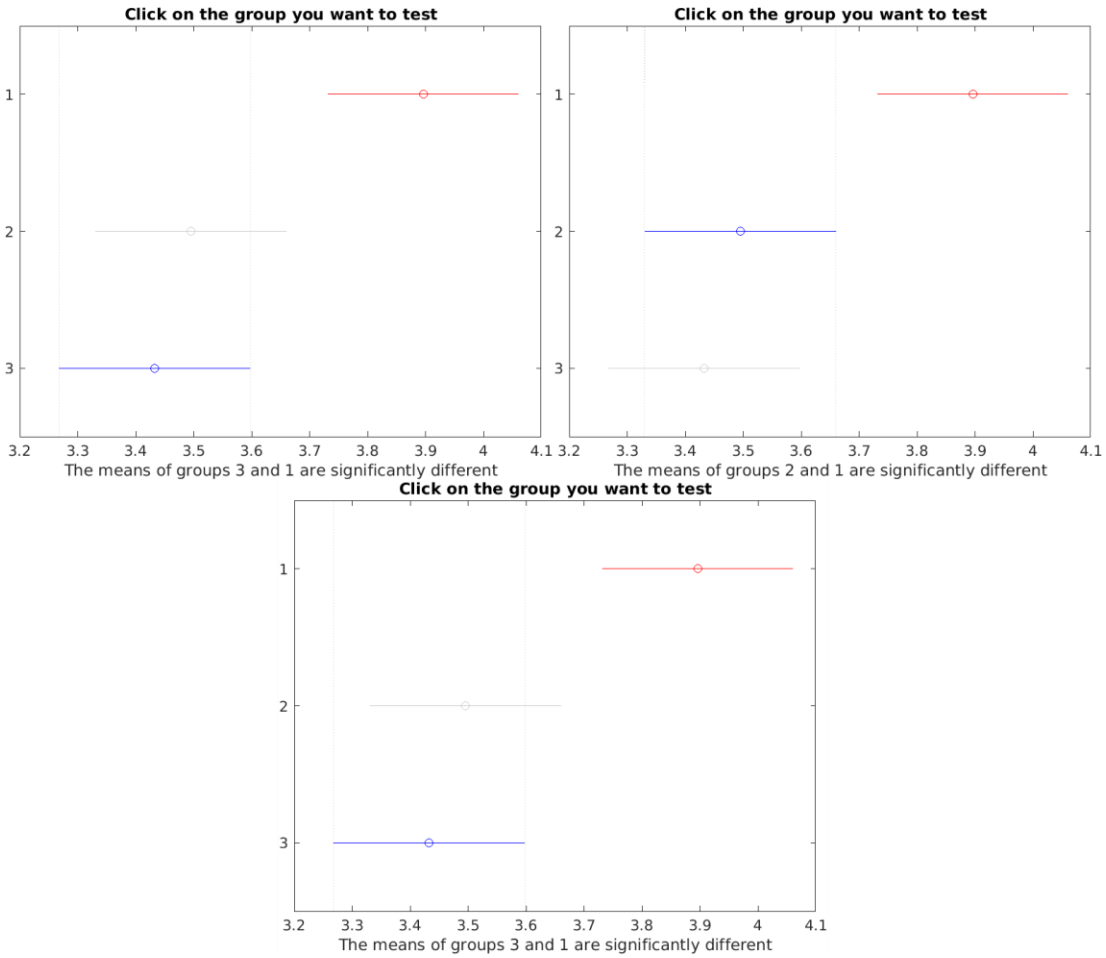
Slika 6.95 - Prikaz Multicompare testa za BRISQUE metodu.

Slike prikazuju da nema značajnih razlika između algoritama.



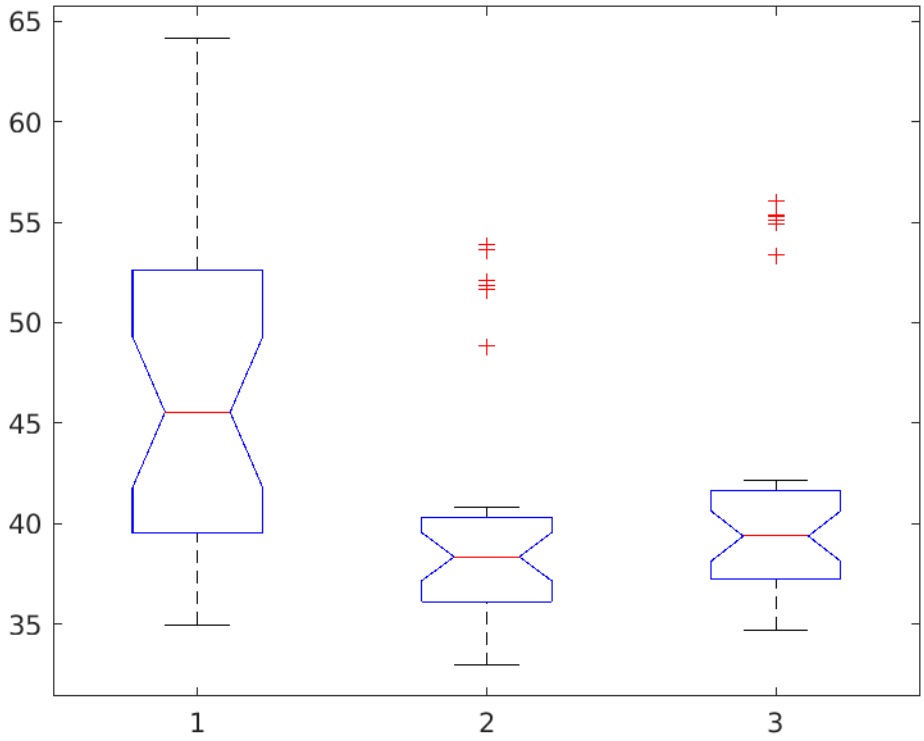
Slika 6.96 - ANOVA dijagram za NIQE metodu.

Prvi algoritam (1) – Medijan = 3.8277, Max = 5, Min = 3.1093
Drugi algoritam (2) – Medijan = 3.3793, Max = 4.4694, Min = 2.5837
Treći algoritam (3) – Medijan = 3.261, Max = 4.5184, Min = 2.6823
F = 6.6, Prob>F = 0.0021



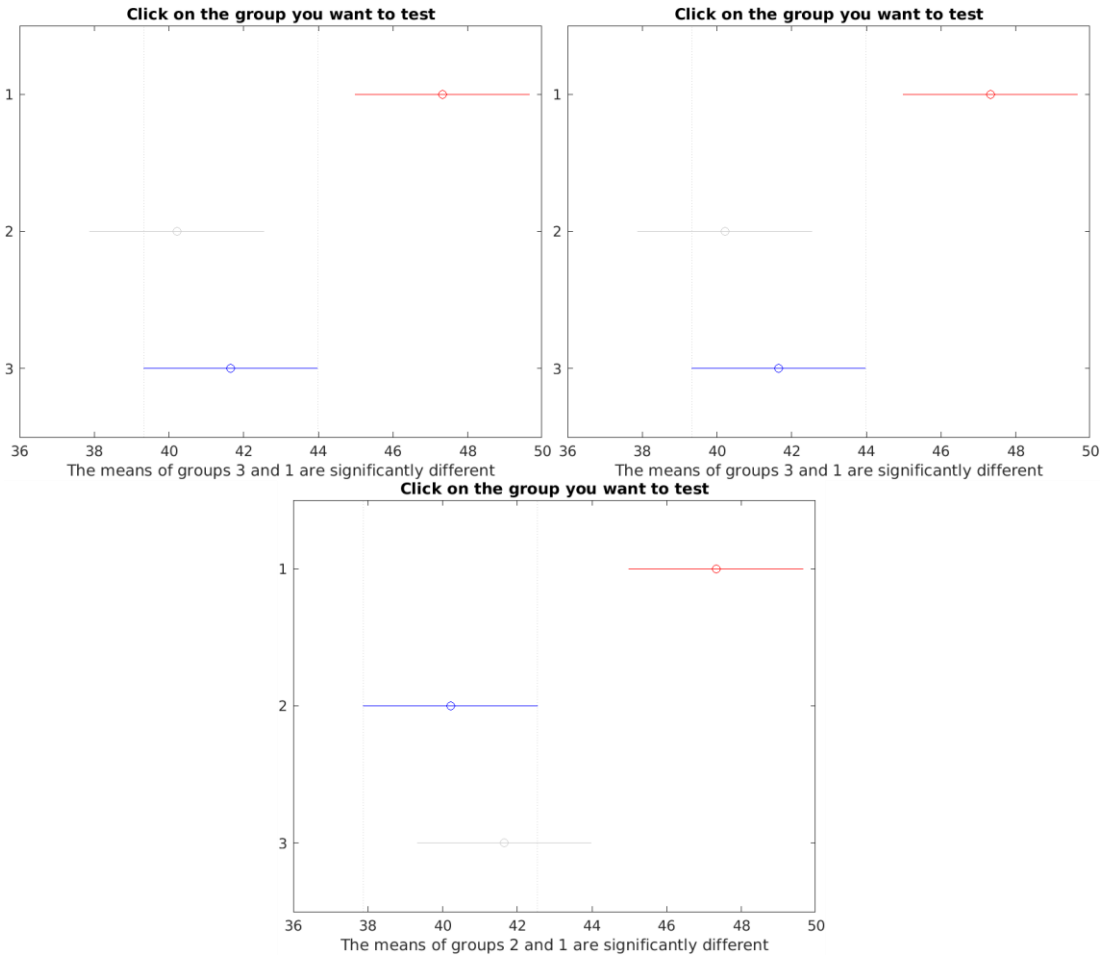
Slika 6.97 - Multicompare test za NIQE metodu.

Multicompare test za NIQE metodu pokazuje da ima značajnih razlika kod prvog i drugog algoritma te kod prvog i trećeg algoritma. Isto se može isčitati i iz ANOVA dijagrama. Rezultat prvog algoritma (Transformacija udaljenosti + Gaussian filter) značajno se razlikuje od ostala dva, kod kojih nema značajnijih razlika.



Slika 6.98 - ANOVA dijagram za PIQE metodu.

Prvi algoritam (1) – Medijan = 45.5473, Max = 64.1536, Min = 34.9741
Drugi algoritam (2) – Medijan = 38.359, Max = 53.8733, Min = 32.9779
Treći algoritam (3) – Medijan = 39.3846, Max = 56.0278, Min = 34.7274
F = 7.37, Prob>F = 0.0011



Slika 6.99 - Multicompare test za PIQE metodu.

Multicompare test za PIQE metodu pokazao je slične rezultate prethodnoj metodi. Iako su brojke kod ove metode znatno veće, to ne znači da se sam rezultat razlikuje. Poput prijašnje metode, drugi i treći algoritam nemaju većih razlika za razliku od prvog koji se značajno razlikuje od njih. Isto se može isčitati iz ANOVA dijagrama na prethodnoj stranici.

6.3.2. Vremenska usporedba popravljanja slike

Svaki od prikazanih algoritama ima svoju brzinu popravljanja slika. Vrijeme potrebno za popravku slike kod svakog algoritma prikazat će se na primjeru jedne od slika. Direktno uspoređivanje vremena nije moguće jer se OpenCV ili Matlab nije koristio za sve algoritme, transformacija udaljenosti + Gaussian filter odrađena je programski (prikazano na početku poglavlja), a Criminisijev i CTBI algoritmi odrađeni su u Matlab-u.

Prvi algoritam, transformacija udaljenosti + Gaussian filter, odrađen je programski u realnom vremenu, te će se napraviti usporedba s ostala dva algoritma kako bi se vidjela vremenska razlika kod algoritma koji radi u realnom vremenu i onih koji ne rade u realnom vremenu.

Slika 6.100 - Korištena slika za vremensku usporedbu.

Vremenska usporedba	s
Transformacija udaljenosti + Gaussian filter	0.025
Criminisijev algoritam	3.037
CTBI algoritam	1.401

Tablica 10 - Vremenska usporedba

7. Zaključak

DIBR tehnike jedne su od ključnih rješenja i obećavajućih alata za podršku naprednih 3D video usluga, sintetiziranjem nekih novih pogleda. Samom ovom rečenicom naglašava se važnost DIBR tehnika za budućnost te upravo ta važnost rezultat je konstantnog razvoja brojnih novih tehnika i unapređenja starijih, jednostavnijih.

Objektivno istraživanje pokazalo je slične rezultate kod NIQE i PIQE metoda, dok se BRISQUE metoda znatno razlikuje. BRISQUE metoda nije pokazala značajnije razlike u rezultatima kod sva tri algoritma, za razliku od druge dvije metode. NIQE i PIQE metode pokazale su da se prvi korišteni algoritam znatno razlikuje od druga dva, koja međusobno nemaju većih razlika. Maksimalne i minimalne vrijednosti svih algoritama ne razlikuju se previše kod BRISQUE metode, te je F vrijednost poprilično niska. Criminisijev i CTBI algoritmi imaju sličnije maksimalne i minimalne vrijednosti kod NIQE i PIQE metoda, dok Transformacija udaljenosti s Gaussian Filterom „odskače“ od njih. NIQE i PIQE metode dale su slične rezultate čime možemo zaključiti da Criminisijev i CTBI algoritam uspješnije popravljaju slike. Iako Transformacija udaljenosti s Gaussian Filterom daje lošije rezultate od ostala dva algoritma, iz vremenske usporedbe možemo vidjeti kako za popravku slika Criminisijev i CTBI algoritam trebaju daleko više vremena. Criminisijev i CTBI algoritam puno uspješnije popravljaju slike, dok Transformacija udaljenosti s Gaussian Filterom iste slike popravlja u realnom vremenu.

Razvojem, kombinacijom, unapređenjem i posebnom prilagodbom programa te izradnje preciznijih i detaljnijih dubinskih mapa, izobličene slike i video zapisi moći će se popraviti daleko bolje, što će dovesti do znatno boljih rezultata, bili oni objektivni ili subjektivni.

U Varaždinu, 07.10.2020. _____

Sveučilište
Sjever

SVEUČILIŠTE
SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, TINA POGAČ (ime i prezime) pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog (obrisati nepotrebno) rada pod naslovom Rekonstrukcija virtualnih pogleda dobivenih pomoću DIBR algoritama (upisati naslov) te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)
Tina Pogač
(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, TINA POGAČ (ime i prezime) neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog (obrisati nepotrebno) rada pod naslovom Rekonstrukcija virtualnih pogleda dobivenih pomoću DIBR algoritama (upisati naslov) čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)
Tina Pogač
(vlastoručni potpis)

8. Literatura

[1] J. Gautier, O. Le Meur, C. Guillemot, "Depth-based image completion for view synthesis", 2011 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), Antalya, svibanj 2011., str. 1-4, doi: 10.1109/3DTV.2011.5877193.

[2] C. Zhu, Y. Zhao, L. Yu, M. Tanimoto, "3D-TV System with Depth-Image-Based Rendering: Architectures, Techniques and Challenges", Springer-Verlag New York, 2013., doi: 10.1007/978-1-4419-9964-1

[3] C. Fehn, "Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV", Proc. SPIE 5291, Stereoscopic Displays and Virtual Reality Systems XI, svibanj 2004., str. 93-104, doi: 10.1117/12.524762

[4] P. A. Assunção, A. Gotchev, "3D Visual Content Creation, Coding and Delivery ",Springer International Publishing, 2019, doi: 10.1007/978-3-319-77842-6

[5] M. Domański et al., "Emerging Imaging Technologies: Trends and Challenges", u: Assunção P., Gotchev A. (eds) 3D Visual Content Creation, Coding and Delivery. Signals and Communication Technology. Springer, Cham, 2019., str. 5-39, doi: 10.1007/978-3-319-77842-6_2

[6] K. Wang, P. An, H. Cheng, H. Li, Z. Zhang, "A New Method of DIBR Based on Background Inpainting", u: Zhang W., Yang X., Xu Z., An P., Liu Q., Lu Y. (eds) Advances on Digital Television and Wireless Multimedia Communications. Communications in Computer and Information Science, vol 331. Springer, Berlin, Heidelberg, 2012., str. 478-484, doi: 10.1007/978-3-642-34595-1_65

[7] S. Tian, L. Zhang, L. Morin, O. Déforges, "A Benchmark of DIBR Synthesized View Quality Assessment Metrics on a New Database for Immersive Media Applications", IEEE Transactions on Multimedia, vol. 21, br. 5, str. 1235-1247, svibanj 2019., doi: 10.1109/TMM.2018.2875307.

[8] S. Takanori, J. Kenji, T. Nobuji, Y. Hiroshi, K. Wegner, "View Synthesis Reference Software (VSRS) 4.2 with improved inpainting and hole filing", ISO/IEC JTC1/SC29/WG11 MPEG2013, M40657, Hobart, Australija, travanj 2017.

[9] A. Mittal, A. K. Moorthy, A. C. Bovik, "No-Reference Image Quality Assessment in the Spatial Domain", IEEE Transactions on Image Processing, vol. 21, br. 12, str. 4695-4708, prosinac 2012., doi: 10.1109/TIP.2012.2214050.

[10] A. Mittal, R. Soundararajan, A. C. Bovik, "Making a “Completely Blind” Image Quality Analyzer", IEEE Signal Processing Letters, vol. 20, br. 3, str. 209-212, ožujak 2013., doi: 10.1109/LSP.2012.2227726.

[11] N. Venkanath, D. Praneeth, M. Chandrasekhar Bh, S. S. Channappayya, S. S. Medasani, "Blind image quality evaluation using perception based features", 2015 Twenty First National Conference on Communications (NCC), Mumbai, ožujak 2015., str. 1-6, doi: 10.1109/NCC.2015.7084843.

9. Popis slika

Slika 2.1 - 2D slika u boji (Europski digitalni TV format) i njen 8-bitna slika dubine s istom prostorno-vremenskom rezolucijom. Prikaz rada i pojašnjenje formule. 13
Slika 4.1 - Primjer 2D fotografije gdje se dubina može lako percipirati. 17
Slika 4.2 - Prikaz svih dubinskih znakova u slikama. 18
Slika 4.3 - Prikaz općenitog modela za 2D-u-3D konverziju. 19
Slika 4.4 - Prikaz dubinskog polja na 2D slici (isječak iz filma Marvel's Avengers - Infinity War, 2018.). 21
Slika 4.5 – Prikaz 3 operacije DIBR procesa 22
Slika 5.1 - Stereo konfiguracija gdje se svi pikseli ne mogu vidjeti iz svih pogleda kamera. 24
Slika 5.2 - Prikaz 3D image warping. 25
Slika 5.3 - Prikaz DIBR izobličene slike, gdje je a) referentna tekstura, b) referentna dubina, c) sintetizirani pogled. 28
Slika 5.4 - Prikaz predprocesiranja dubine prije DIBR-a 29
Slika 5.5 - Prikaz uklanjanja dijela slike te popunjavanje veće praznine koristeći Criminsijev algoritam. 32
Slika 5.6 - Prikaz Criminisijevog dijagrama 33
Slika 5.7 - Prikaz dijagrama notacije 34
Slika 5.8 - Prikaz VSRS modela 35
Slika 6.1 - Sadržaj programa. 44
Slika 6.2 - Prikaz rada programa. 44
Slika 6.3 - Prikaz originalnih korištenih slika i njihovih dubinskih mapa. 45
Slika 6.4 - Prikaz prve slike s pogledom u desno (Gaussian). 46
Slika 6.5 - Prikaz prve slike s pogledom u lijevo (Gaussian). 46
Slika 6.6 - Prikaz prve slike s pogledom prema gore (Gaussian). 47
Slika 6.7 - Prikaz prve slike s pogledom prema dolje (Gaussian). 48
Slika 6.8 - Prikaz prve slike s pogledom u gornji desni kut (Gaussian). 48
Slika 6.9 - Prikaz prve slike s pogledom u donji lijevi kut (Gaussian). 49
Slika 6.10 - Prikaz druge slike s pogleodm u desno (Gaussian). 49
Slika 6.11 - Prikaz druge slike s pogledom u lijevo (Gaussian). 50
Slika 6.12 - Prikaz druge slike s pogledom prema gore (Gaussian). 50
Slika 6.13 - Prikaz druge slike s pogledom prema dolje (Gaussian). 51
Slika 6.14 - Prikaz druge slike s pogledom u gornji desni kut (Gaussian). 51
Slika 6.15 - Prikaz druge slike s pogledom u donji lijevi kut (Gaussian). 52
Slika 6.16 - Prikaz treće slike s pogledom u desno (Gaussian). 52
Slika 6.17 - Prikaz treće slike s pogledom u lijevo (Gaussian). 52
Slika 6.18 - Prikaz treće slike s pogledom prema gore (Gaussian). 53
Slika 6.19 - Prikaz treće slike s pogledom prema dolje (Gaussian). 53
Slika 6.20 - Prikaz treće slike s pogledom u gornji desni kut (Gaussian). 53

Slika 6.21 - Prikaz treće slike s pogledom u donji lijevi kut (Gaussian). 54
Slika 6.22 - Prikaz četvrte slike s pogledom u desno (Gaussian). 54
Slika 6.23 - Prikaz četvrte slike s pogledom u lijevo (Gaussian). 54
Slika 6.24 - Prikaz četvrte slike s pogledom prema gore (Gaussian). 55
Slika 6.25 - Prikaz četvrte slike s pogledom prema dolje (Gaussian). 55
Slika 6.26 - Prikaz četvrte slike s pogledom u gornji desni kut (Gaussian). 55
Slika 6.27 - Prikaz četvrte slike s pogleodm u donji lijevi kut (Gaussian). 56
Slika 6.28 - Prikaz pete slike s pogledom u desno (Gaussian). 56
Slika 6.29 - Prikaz pete slike s pogledom u lijevo (Gaussian). 56
Slika 6.30 - Prikaz pete slike s pogledom prema gore (Gaussian). 57
Slika 6.31 - Prikaz pete slike s pogledom prema dolje (Gaussian). 57
Slika 6.32 - Prikaz pete slike s pogledom u gornji desni kut (Gaussian). 57
Slika 6.33 - Prikaz pete slike s pogledom u donji lijevi kut (Gaussian). 58
Slika 6.34 - Prikaz prve slike s pogledom u desno (Criminisi). 59
Slika 6.35 - Prikaz prve slike s pogledom u lijevo (Criminisi). 60
Slika 6.36 - Prikaz prve slike s pogledom prema gore (Criminisi). 60
Slika 6.37 - Prikaz prve slike s pogledom prema dolje (Criminisi). 61
Slika 6.38 - Prikaz prve slike s pogledom u gornji desni kut (Criminisi). 61
Slika 6.39 - Prikaz prve slike s pogledom u donji lijevi kut (Criminisi). 62
Slika 6.40 - Prikaz druge slike s pogledom u desno (Criminisi). 62
Slika 6.41 - Prikaz druge slike s pogledom u lijevo (Criminisi). 63
Slika 6.42 - Prikaz druge slike s pogledom prema gore (Criminisi). 63
Slika 6.43 - Prikaz druge slike s pogleodm prema dolje (Criminisi). 63
Slika 6.44 - Prikaz druge slike s pogledom u gornji desni kut (Criminisi). 64
Slika 6.45 - Prikaz druge slike s pogledom u donji lijevi kut (Criminisi). 64
Slika 6.46 - Prikaz treće slike s pogledom u desno (Criminisi). 64
Slika 6.47 - Prikaz treće slike s pogledom u lijevo (Criminisi). 65
Slika 6.48 - Prikaz treće slike s pogledom prema gore (Criminisi). 65
Slika 6.49 - Prikaz treće slike s pogledom prema dolje (Criminisi). 65
Slika 6.50 - Prikaz treće slike s pogledom u gornji desni kut (Criminisi). 65
Slika 6.51 - Prikaz treće slike s pogledom u donji lijevi kut (Criminisi). 66
Slika 6.52 - Prikaz četvrte slike s pogledom u desno (Criminisi). 66
Slika 6.53 - Prikaz četvrte slike s pogledom u lijevo (Criminisi). 66
Slika 6.54 - Prikaz četvrte slike s pogledom prema gore (Criminisi). 67
Slika 6.55 - Prikaz četvrte slike s pogledom prema dolje (Criminisi). 67
Slika 6.56 - Prikaz četvrte slike s pogledom u gornji desni kut (Criminisi). 67
Slika 6.57 - Prikaz četvrte slike s pogledom u donji lijevi kut (Criminisi). 67
Slika 6.58 - Prikaz pete slike s pogledom u desno (Criminisi). 68
Slika 6.59 - Prikaz pete slike s pogledom u lijevo (Criminisi). 68
Slika 6.60 - Prikaz pete slike s pogledom prema gore (Criminisi). 68
Slika 6.61 - Prikaz pete slike s pogledom prema dolje (Criminisi). 69
Slika 6.62 - Prikaz pete slike s pogledom u gornji desni kut (Criminisi). 69

Slika 6.63 - Prikaz pete slike s pogledom u donji lijevi kut (Criminisi). 69
Slika 6.64 - Prikaz prve slike s pogledom u desno (CTBI). 70
Slika 6.65 - Prikaz prve slike s pogledom u lijevo (CTBI). 71
Slika 6.66 - Prikaz prve slike s pogledom prema gore (CTBI). 71
Slika 6.67 - Prikaz prve slike s pogledom prema dolje (CTBI). 72
Slika 6.68 - Prikaz prve slike s pogledom u gornji desni kut (CTBI). 73
Slika 6.69 - Prikaz prve slike s pogledom u donji lijevi kut (CTBI). 73
Slika 6.70 - Prikaz druge slike s pogledom u desno (CTBI). 74
Slika 6.71 - Prikaz druge slike s pogledom u lijevo (CTBI). 74
Slika 6.72 - Prikaz druge slike s pogledom prema gore (CTBI). 74
Slika 6.73 - Prikaz druge slike s pogledom prema dolje (CTBI). 75
Slika 6.74 - Prikaz druge slike s pogledom u gornji desni kut (CTBI). 75
Slika 6.75 - Prikaz druge slike s pogledom u donji lijevi kut (CTBI). 75
Slika 6.76 - Prikaz treće slike s pogledom u desno (CTBI). 76
Slika 6.77 - Prikaz treće slike s pogledom u lijevo (CTBI). 76
Slika 6.78 - Prikaz treće slike s pogledom prema gore (CTBI). 76
Slika 6.79 - Prikaz treće slike s pogledom prema dolje (CTBI). 76
Slika 6.80 - Prikaz treće slike s pogledom u gornji desni kut (CTBI). 77
Slika 6.81 - Prikaz treće slike s pogledom u donji lijevi kut (CTBI). 77
Slika 6.82 - Prikaz četvrte slike s pogledom u desno (CTBI). 77
Slika 6.83 - Prikaz četvrte slike s pogledom u lijevo (CTBI). 77
Slika 6.84 - Prikaz četvrte slike s pogledom prema gore (CTBI). 78
Slika 6.85 - Prikaz četvrte slike s pogledom prema dolje (CTBI). 78
Slika 6.86 - Prikaz četvrte slike s pogledom u gornji desni kut (CTBI). 78
Slika 6.87 - Prikaz četvrte slike s pogledom u donji lijevi kut (CTBI). 78
Slika 6.88 - Prikaz pete slike s pogledom u desno (CTBI). 79
Slika 6.89 - Prikaz pete slike s pogledom u lijevo (CTBI). 79
Slika 6.90 - Prikaz pete slike s pogledom prema gore (CTBI). 79
Slika 6.91 - Prikaz pete slike s pogledom prema dolje (CTBI). 80
Slika 6.92 - Prikaz pete slike s pogledom u gornji desni kut (CTBI). 80
Slika 6.93 - Prikaz pete slike s pogleodm u donji lijevi kut (CTBI). 80
Slika 6.94 - Prikaz ANOVA dijagrama za BRISQUE metodu. 87
Slika 6.95 - Prikaz Multicompare testa za BRISQUE metodu. 88
Slika 6.96 - ANOVA dijagram za NIQE metodu. 89
Slika 6.97 - Multicompare test za NIQE metodu. 90
Slika 6.98 - ANOVA dijagram za PIQE metodu. 91
Slika 6.99 - Multicompare test za PIQE metodu. 92
Slika 6.100 - Korištena slika za vremensku usporedbu. 93

10. Popis tablica

Tablica 1 - BRISQUE metoda - transformacija udaljenosti + Gaussian filter 81
Tablica 2 - BRISQUE metoda - Criminisi 81
Tablica 3 – BRISQUE metoda - CTBI 82
Tablica 4 - NIQE metoda - transformacija udaljenosti + Gaussian filter 83
Tablica 5 - NIQE metoda - Criminisi 83
Tablica 6 - NIQE metoda - CTBI 84
Tablica 7 - PIQE metoda - Transformacija udaljenosti + Gaussian filter 85
Tablica 8 - PIQE metoda - Criminisi 85
Tablica 9 - PIQE metoda - CTBI 86