



Univerzitet u Beogradu - Elektrotehnički fakultet

Katedra za signale i sisteme



MASTER RAD

Višestruki pristup spajanja slika planarne i cilindrične projekcije u panorame

Kandidat

dipl. inž. Uroš Petković, broj indeksa 2020/3027

Mentor

dr Veljko Papić, prof.

Beograd, *septembar* 2021. godine

PREDGOVOR

Master rad baziran je na primenjivanju znanja stečenog na predmetu kompjuterska vizija, sa šifrom predmeta 13M051KV pod vođstvom profesora dr Veljka Papića, pohađanog u drugom semestru master akademskih studija. Rad nosi naziv "Višestruki pristup spajanja slika planarne i cilindrične projekcije u panorame" u čijoj izradi je učestvovao student Uroš Petković, diplomirani inženjer elektrotehnike i računarstva. Ovaj rad predstavlja jednu od mogućih implementacija algoritama za spajanje slika koje je danas široko rasprostranjeno i koristi se u svakom naprednom telefonskom uređaju, a i mnogim drugim oblastima kompjuterske vizije.

Izrada master rada omogućena je samostalnim prikupljanjem adekvatnog materijala za implementaciju algoritma, odnosno fotografisanjem odgovarajućih pejzaža koji će obuhvatiti sve tipove slika koji se mogu naći u svakodnevnoj upotrebi. Set se sastoji od 4 direktorijuma slika gde prvi direktorijum predstavlja slike u kojima se javljaju kućni element i nameštaj, drugi direktorijum predstavlja slike u kojima se javljaju motivi zgrada, kuća i prirode, treći direktorijum predstavlja slike koje prirode najvećim delom, dok četvrti direktorijum predstavlja slike prirode i okoline za pravljenje panorama u 360 stepeni sa ukupno 30 slika od kojih će biti sastavljene četiri pojedinačne panorame. Praktični deo master rada i implementacija samog rešenja rađena je samostalno od strane studenta, uz nesebičnu pomoć mentora.

REZIME RADA

U ovom master radu zadatak je bio da se na različite načine izvrši spajanje prikupljenih slika predstavljenih u drugačijim koordinatnim sistemima, planarnom i cilindričnom, kao i upoređivanje rezultata dobijenih ovom obradom. Postoji više načina rešavanja ovog problema, stoga je potrebno imati znanje iz drugih oblasti, kao što su digitalna obrada slike i prepoznavanje oblika, kao i osnova mašinskog učenja. Najpre se od dostupnog seta slika različitim algoritmima izvrši izdvajanje odgovarajućih obeležja koja će se koristiti u daljoj obradi, tačnije pronalaskom karakterističnih Harisovih tačaka i njihovom obradom, kao i korišćenjem ostalih algoritama kao što su *SIFT (Scale invariant feature transform)*, *SURF (Speeded-Up Robust Features)* i *ORB (Oriented FAST and Rotated BRIEF)* algoritmi. Nakon toga nalaze se parovi poklapanja odgovarajućih obeležja na osnovu lokalnih maksimuma, gde se odbacuju *outlier* obeležja pomoću *RANSAC* algoritma, odnosno ona obeležja koja ne odgovaraju datom poklapanju slika, gde se na kraju od preostalih obeležja vrši spajanje slika u panoramu pronalaskom adekvatne homografije. Prilikom spajanja slika radi se *blending* slike na porubima, odnosno spojevima slika kako bi se što manje uočio efekat spajanja. Tako dobijene panorame se zatim poravnavaju i seku, čime se dobijaju konačne panorame. Ovakav algoritam obrade slika je danas široko rasprostranjen u primeni u oblasti kompjuterske vizije i njoj srodnih oblasti, kako u svakodnevnoj, tako i u profesionalnoj upotrebi.

ZAHVALNICA

Posebna zahvalnost ističe se mentoru dr Veljku Papiću na pomoći i izdvojenom vremenu kada su u pitanju saveti i konsultacije u vezi rešavanja zadatog problema ovog master rada, kao i snabdevanju potrebnom literaturom za uspešno realizovanje. Pored toga, veliku zahvalnost dugujem svojoj porodici koja mi je pomogla u prikupljanju materijala za izradu master rada, kao i nesebičnoj podršci koju su mi pružali prilikom svakog mog novog životnog podviga.

dipl. inž. Uroš Petković

U Beogradu, *septembar* 2021. godine

SADRŽAJ

PREDGOVOR	2
REZIME RADA	3
ZAHVALNICA	4
SADRŽAJ	5
1 UVOD	6
2 OSNOVI OBRADE I SPAJANJA SLIKA	10
2.1 Planarne i cilindrične projekcije slika	10
2.2 Ekstrakcija obeležja pomoću Harisovih tačaka	12
2.3 <i>SIFT</i> algoritam za ekstrakciju obeležja	14
2.4 <i>SURF</i> algoritam za ekstrakciju obeležja	15
2.5 <i>ORB</i> algoritam za ekstrakciju obeležja	18
2.6 Algoritam za poklapanje parova obeležja	20
2.7 <i>RANSAC</i> algoritam i procena homografije	21
2.8 Algoritmi <i>blending</i> -a, poravnavanja i isecanja panorame	23
3 METODOLOGIJA RADA	25
4 IMPLEMENTACIJA I REZULTATI	28
5 DISKUSIJA	48
6 ZAKLJUČAK	50
7 LITERATURA	51
PRILOG A	52

1 UVOD

Kompjuterska vizija je interdisciplinarna nauka čiji je zadatak da nauči računar kako da postigne visok nivo razumevanja digitalne slike i videa. Sa tačke gledišta inženjerstva, ona nastoji da automatizuje zadatke koje vizuelni sistem čoveka može da izvršava. Zadaci kompjuterske vizije uključuju metode za prikupljanje, obradu, analizu i razumevanje digitalne slike, kao i izdvajanje višedimenzionalnih podataka iz realnog sveta u cilju da se dobiju numeričke ili simboličke informacije. Razumevanje u ovom kontekstu znači pretvaranje vizuelne slike (ulaz mrežnjače) u opis sveta koji može da komunicira sa drugim misaonim procesima i izaziva odgovarajuće akcije. Ovakvo razumevanje slika se može posmatrati kao raspletanje simboličkih informacija koji se nalaze u slici, koristeći modele izgrađene uz pomoć geometrije, fizike, statistike i teorije učenja. Kao naučna disciplina, kompjuterska vizija se bavi teorijom veštačkih sistema koja izdvaja informacije iz slike. Podaci o slici mogu imati različite oblike, kao što su video zapisi, pogled iz više kamera, ili višedimenzionalni podaci medicinskog skenera. Kao tehnološka disciplina, kompjuterska vizija nastoji da primeni teorije i modele za izgradnju sistema kompjuterskog vida. Pod-domeni kompjuterske vizije uključuju rekonstrukcije scena, detekciju događaja, video-praćenje, prepoznavanje objekata, 3D procenu pozicije, učenje, indeksiranje, procenu pokreta i restauraciju slike, spajanje slike i mnoge druge oblasti.

Spajanje slika predstavlja jedan od algoritama koji se dosta upotrebljava u oblastima kompjuterske vizije, zajedno sa ostalim algoritmima za detekciju, praćenje i prepoznavanje određenih pokreta i pojava na slikama i video zapisima. Primena ovog i njemu sičnih algoritama može se ogledati u njihovom korišćenju u naprednim telefonskim uređajima koji u svojoj funkciji imaju pravljenje panoramnih slika, snimanju geografskih lokacija iz različitih perspektiva i koordinatnih sistema, izradi online mapa za prikaz i navigaciju, prikaz slika u krugu od 360 stepeni i mnogim drugim sferama. Tako spojene slike se nazivaju panorame, a njena definicija kaže da one predstavljaju celokupan izgled svih predmeta u prirodi koji se mogu obuhvatiti pogledom sa izvesne tačke i umetničko prikazivanje takvog izgleda u obliku okrugle ili dugačke slike, a u novije vreme obično se koristi u svrhu prikaza udešenih prostorija i ostalih arhitektonskih radova. Kompjuterska vizija je relativno mlada nauka, stoga je poslednjih godina sa razvojem tehnologije ona doživela još veću ekspanziju i počela je sve više da se koristi. U ovom radu akcenat će biti baziran na višestrukom pristupu spajanju slika planarne i cilindrične projekcije u panorame i upoređivanju dobijenih rezultata danas aktuelnih rešenja ovog problema. Na raspolaganju se nalazi 30 odabranih slika pomoću kojih će se izvršiti dato spajanje. Slike su podeljene u četiri direktorijuma po onome šta prikazuju, a to su kućni elementi i nameštaj, eksterijer sa izraženim arhitektonskim građevinama i

prirodom, kao i slike koje predstavljaju pretežno prirodu. Prvi korak je izdvajanje obeležja iz datih slika. Kako se ovaj rad bazira na izradi više algoritama za izdvajanje obeležja, algoritmi koji će se posmatrati su: algoritam za izdvajanje karakterističnih Harisovih tačaka i izdvajanje obeležja na osnovu lokalnih maksimuma, *SIFT (Scale invariant feature transform)*, *SURF (Speeded-Up Robust Features)* i *ORB (Oriented FAST and Rotated BRIEF)* algoritmi. Nakon toga se od dobijenih obeležja na svakoj slici, za svaki par slika nalaze poklapajući parovi obeležja na osnovu njihove sličnosti pomoću kojih će se izvršiti pomeranje i spajanje slika. Kako je ovo realan problem i sva pronađena obeležja nisu adekvatna, postoje ona obeležja koja su loše procenjena i koja bi loše uticala na sam algoritam. Zato se nakon ovoga koristi *RANSAC* algoritam za uklanjanje loše odabranih obeležja. Nakon toga se vrši spajanje slika pronalaskom odgovarajuće homografije, odnosno adekvatnog pomeranja po osama, kako bi se slike što bolje poklopile. Prilikom spajanja slika vrši se *blending* slike na porubima, odnosno spojevima slika, kako bi se što manje mogao primetiti uticaj spajanja slika i kako bi se dobili pikseli približnih intenziteta koji će odavati utisak savršeno spojene slike. Na samom kraju, takve panorame planarne i cilindrične projekcije poravnavaju se i seku, čime se dobijaju konačni rezultati koji će kasnije biti detaljno opisani. Kako je ovo jedna od jako popularnih tema današnjice, kao i kompjuterska vizija uopšte, jako je širok spektar literature koja se može naći kada je u pitanju ovaj i njemu slični algoritmi za obradu slika.

[1] Matthew Brown i David G. Lowe sa univerziteta u Britanskoj Kolumbiji u Vankuveru, Kanadi, su u svom radu *"Automatic Panoramic Image Stitching using Invariant Features"* prikazali sličan pristup rešavanju ovog problema. Njihov rad se odnosi na potpuno automatizovano spajanje panoramskih slika. Prethodni pristupi imali su restrikcije kada je u pitanju redosled slika u cilju uspostavljanja odgovarajuće slike. U njihovom radu problem su formulisali kao problem podudaranja više slika koristeći invarijantna lokalna obeležja kako bi pronašli podudaranja između svih slika, stoga je njihova metoda neosetljiva na redosled slika, orientaciju, skaliranje, osvetljenost i pristustvo šuma na slikama. Prvi korak u njihovom algoritmu predstavlja izdvajanje obeležja *SIFT* algoritmom i pronađenje podudaranja dobijenih obeležja na svim slikama. Obeležja ovog algoritma nalaze se na maksimumima i minimumima *DoG* funkcije (*Difference of Gaussian function*) gde su svakom obeležju dodeljeni karakteristična orientacija i skaliranje gde se zapravo invarijantni deskriptor izračunava akumuliranjem lokalnih gradijenata u orientacionim histogramima. Ovo omogućava ivicama da se lagano pomeraju bez promene vektora deskriptora dajući na robusnosti. Budući da su *SIFT* obeležja invarijantna na rotaciju i skaliranje, to omogućava algoritmu da obrađuje slike različite orientacije i zuma što ne bi bilo moguće ako bi se koristili neki alternativni algoritmi kao što je upotreba Harisovih tačaka. Nakon ekstrakcije obeležja, primenom metode *KNN (K Nearest Neighbours)* nalazi se k najbližih suseda obeležja za svaku sliku i nalaze parovi obeležja. Potom se izaberu one dve slike koje imaju najviše pronađenih parova poklapanja. Nakon toga se primenom *RANSAC* algoritma odbacuju *outlier*-i, nakon čega se vrši verifikacija parova probabilističkim modelom. Za svaki par *inlier*-a i *outlier*-a određuje se i upoređuje

verovatnoća da li dati par pripada adekvatnom skupu ili ne, odnosno da li su prepoznati kao dobro ili kao loše poklapanje slike. Potom se vrši *Bundle Adjustment*. Slike se dodaju u podešavač snopa jedna po jedna, sa slikom sa kojom se najbolje podudara (ima maksimalan broj dobrih poklapanja) koje se dodaju u svakom koraku. Nova slika se inicijalizuje istom rotacijom i žižnom daljinom kao slika kojoj se najbolje podudara, nakon čega se parametri ažuriraju Levenberg-Marquardt algoritmom. Na samom kraju, slike se spajaju u konačnu panoramu nakon čega se vrši ispravljanje slike i *Multi-Band Blending* slike kako bi se uklonili tragovi lepljenja slika čime se dobijaju konačni rezultati.

[2] *Natthavut Vong* i *Chatklaw Jareonpona* sa Mahasarakham univerziteta u Tajlandu su u svom radu "*Multi image stitching with cylindrical surface base on local feature matching for solving the distortion problem*" predstavili jedan od načina spajanja slika u cilindričnoj projekciji. Sam algoritam podeljen je u tri celine. Prvo se radi ekstrakcija i poklapanje obeležja. Položaj preklapajućeg područja izvlači se iz karakteristika slike kao što su boja, gustina piksela itd. Za izdvajanje obeležja korišćen je *SURF* algoritam koji je brz i veoma otporan na promene u slici kao što su rotacija, skaliranje, zamućenje i osvetljenost. Ovaj algoritam vrši detekciju na osnovu drugog izvoda Gausa i opisuje ih koristeći Haar talasiće što zahteva kraće vreme izračunavanja. Izlaz koji se naziva ključnim tačkama identificuje lokaciju, skaliranje i orijentaciju. Sličnost između slika računata je Euklidovom distancicom gde mala vrednost distance govori o tome da su tačke slične. Nakon toga radi se estimacija transformacione matrice i uvijanja pomoću *RANSAC* algoritma. Prednost ovog metoda je robusna estimacija matrice homografije i odgovorajućih ključnih tačaka. Nakon toga, uvijanje slika se koristi za spajanje slika u panoramu. Kako uvijanje slika u planarnom sistemu, kada je u pitanju veći broj slika, može dovesti do pojave distorzija, ovaj rad se bazira na uvijanju slika u cilindričnom sistemu gde se dobijaju konačne panorame. Kao nedostatak ovog rada jesu slike na kojima nije izvršen *blending*, pa se date pozicije na samom porubu slika mogu jasno uočiti.

[3] *Richard Szeliski* i *Heung-Yeung Shum* su u svom istraživačkom radu pod nazivom "*Creating Full View Panoramic Image Mosaics and Environment Maps*" govorili su o pristupima rešavanju i prikazivanju panorama. Takođe, pominju se i različite vrste projekcija, kao i njihove osobine, prednosti i mane. Izložen je postupak mapiranja planarnih slika u slike cilindrične i sferične projekcije sa detaljnim postupkom i objašnjenjima, kao i algoritmima za izdvajanje obeležja. Pored toga, dato je više načina kojima se može tačno estimirati žižna daljina koja je od ključne važnosti kada je u pitanju prebacivanje slika u cilindrični i sferični sistem, kao i pojašnjenja kada je u pitanju poravnanje i *blending* slike.

Pored ovih metoda, u literaturi se mogu naći i mnoge druge implementacije algoritama na bazi spajanja slika, sudeći po tome da se ova oblast neprestano širi i razvija i stalno izviru nove ideje, pristupi problemima, kao i sve bolja i priznatija rešenja.

Cilj ovog rada je implementirati takav algoritam koji će uspešno rešavati problem spajanja slika na više načina i to u različitim projekcijama, sa različitim tipovima prosleđenih obeležja. Primena ovog algoritma mogla bi se ogledati u njihovom korišćenju u naprednim telefonskim uređajima koji u svojoj funkciji imaju pravljenje panoramskih slika, snimanju geografskih lokacija iz različitih perspektiva i koordinatnih sistema, izradi online mapa za prikaz i navigaciju, prikaz slika u krugu od 360 stepeni i mnogim drugim sferama. Ono na čemu leži osnova ovog algoritma biće opisano u sledećim poglavljima. Najpre, u poglavlju Osnovi obrade i spajanja slika biće date teorijske osnove koje se tiču same obrade slika i njenog spajanja, objašnjenja svih algoritama koji se koriste u radu, kao što su *SIFT (Scale invariant feature transform)*, *SURF (Speeded-Up Robust Features)* i *ORB (Oriented FAST and Rotated BRIEF)* algoritmi, izdvajanje karakterističnih Harisovih tačaka, algoritam za mapiranje slika u cilindrični sistem, *RANSAC* algoritam za uklanjanje outlier-a i pronalaženje idealne homografije, kao i algoritmi za spajanje, poravnanje i isecanje datih slika. Nakon toga, u poglavlju Metodologija rada prethodno definisani algoritmi obrade slika kombinuju se u jedan sklop koji čini srž rešenja, a čiji će delovi biti pojedinačno detaljno objašnjeni. U sekciji implementacija i rezultati biće prikazani rezultati koji su dobijeni implementiranjem datog rešenja, kao i njihovo upoređivanje. Na samom kraju, u poglavljima Diskusija i Zaključak biće izložene činjenice koje se odnose na date rezultate, kvalitet rešenja, poređenje sa postojećom literaturom, moguće nadogradnje algoritma, kao i konačni sud o ovom radu.

2 OSNOVI OBRADE I SPAJANJA SLIKA

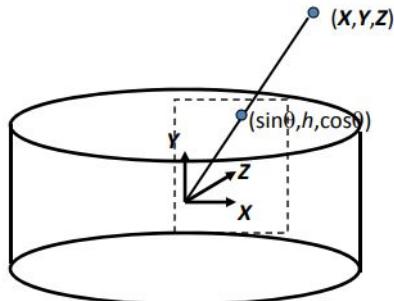
2.1 Planarne i cilindrične projekcije slika

Slike mogu biti prikazane u različitim koordinatnim sistemima. Slike sa kojima se najčešće susrećemo su one koje su prikazane u planarnoj projekciji i to u *RGB* kolor sistemu u formatu $M \times N \times 3$ gde M i N predstavljaju širinu i visinu slike, odnosno broj piksela po širini i visini slike, dok broj 3 predstavlja broj različitih plejnova koji u ovom slučaju odgovara plejnovima R , G i B . U ovom radu bavićemo se spajanjem više slika u jednu panoramu i to takvih da sve slike imaju isti centar projekcije što je omogućeno upotrebom stalka za fotografisanje. Kako u ovom procesu učestvuje veliki broj fotografija (na najvećoj panorami 7 fotografija, a može i više), prilikom spajanja velikog broja slika planarne projekcije može doći do pojave distorzija na slici, odnosno ne tako dobrih poklapanja na mestima gde se slike spajaju, a što se najviše ogleda u tome kada su u pitanju oštiri ili ravni elementi u slici, kao što su građevine, stilski nameštaj, enterijer i ostali mogući predmeti koji mogu dovesti do neželjenih pojava. Zato se pored ovog standardnog koordinatnog sistema koriste i drugi sistemi poput cilindričnog i sfernog koordinatnog sistema. Ovi sistemi su široko rasprostranjeni i primenljivi kada su u pitanju razni prikazi, od arhitektonskih modela do geografskih prikaza i njima sličnim projekcijama. U ovom radu, pored planarnog sistema akcenat će biti i na cilindričnom koordinatnom sistemu. Zadatak je reprojektovati sve slike u cilindričnu projekciju i spojiti ih u navedenu panoramu. Za ovakav postupak potrebno je poznavati žižne daljine slika. Žižne daljine moguće je estimirati različitim načinima. Može se inicijalizovati iz homografije H , može se aproksimirati na osnovu karakteristika kamere i slike, izračunati ručnim merenjem, odnosno korišćenjem trake zadatih dužina i slično. U ovom slučaju žižna daljina estimirana je na osnovu karakteristika kamere. Korišćen je Nikon D5200 DSLR fotoaparat sa prenosivim objektivom Nikkor 18-140mm f/3.5-5.6G ED VR čije specifikacije se mogu naći na internetu. Kako je datim objektivom moguće podešavati žižnu daljinu, u ovom slučaju izabrana je najmanja žižna daljina, odnosno najmanji otvor blende, pa žižna daljina u milimetrima iznosi 18 mm. Korišćen je mod za slikanje u kojem su dimenzije slike 6000x4000 odakle se žižna daljina može estimirati sledećom formulom:

$$f(px) = \frac{\text{širina slike}(px)}{\text{širina senzora}(mm)} * f(mm) = \frac{6000px}{97mm} * 18mm \approx 1115px \quad [1]$$

Kako su ove dimenzije slika prevelike i zahtevaju jako veliko vreme izračunavanja, na datim slikama izvršeno je smanjivanje rezolucije, pa su konačne dimenzije slika za obradu 1200x800.

Sada kada je poznata žična duljina slika, potrebno je slike prebaciti u cilindrični sistem. To se može uraditi na sledeći način:



$$\theta = \frac{x - x_c}{f} \quad h = \frac{y - y_c}{f} \quad [2]$$

$$\hat{x} = \sin \theta \quad \hat{y} = h \quad \hat{z} = \cos \theta \quad [3]$$

$$x_{cil} = \frac{f\hat{x}}{\hat{z}} + x_c \quad y_{cil} = \frac{f\hat{y}}{\hat{z}} + y_c \quad [4]$$

Slika 1 – Prikaz cilindrične projekcije.

Slika preuzeta iz [4]

Ovim postupkom iz planarnog sistema prelazimo u cilindrični sistem koji će se koristiti nadalje, zajedno sa planarnim sistemom. Primeri slika u planarnom i cilindričnom koordinatnom sistemu mogu se naći na slikama ispod.



a)



b)

Slika 2 – Slike korišćene i dobijene u radu; a) Planarna projekcija; b) Cilindrična projekcija

2.2 Ekstrakcija obeležja pomoću Harisovih tačaka

Prvi od algoritama koji će se razmatrati u ovom radu jeste algoritam dobijanja Harisovih tačaka i pronalazak obeležja iz tako definisanih tačaka koji se zasniva na ideji istraživača pod imenom Chris Harris koji je ovu tezu izložio još davne 1988. godine. On se bazira na pronalaženju čoškova na slikama koji imaju ključnu osobinu da u regionu oko čoškova gradijent može imati dva ili više dominantnih pravaca. Ove tačke na slikama mogu se lako pronaći ako se posmatraju kroz neku prozorsku funkciju. Pomeranje dovoljno malog prozora u bilo kom smeru trebalo bi da rezultuje promenama u osvetljaju. Ako bismo prozorom naišli na ivicu slike, koja predstavlja horizontalnu ili vertikalnu promenu, ne bi bilo promena u pravcu ivice i zato se to ne smatra čoškom u slici, dok kada je u pitanju čošak imamo značajne promene u svim pravcima. Promena intenziteta za neki pomeraj $[u, v]$ može se definisati na sledeći način:

$$E(u, v) = \sum_{x,y} w(x, y)[I(x + u, y + v) - I(x, y)]^2 \quad [5]$$

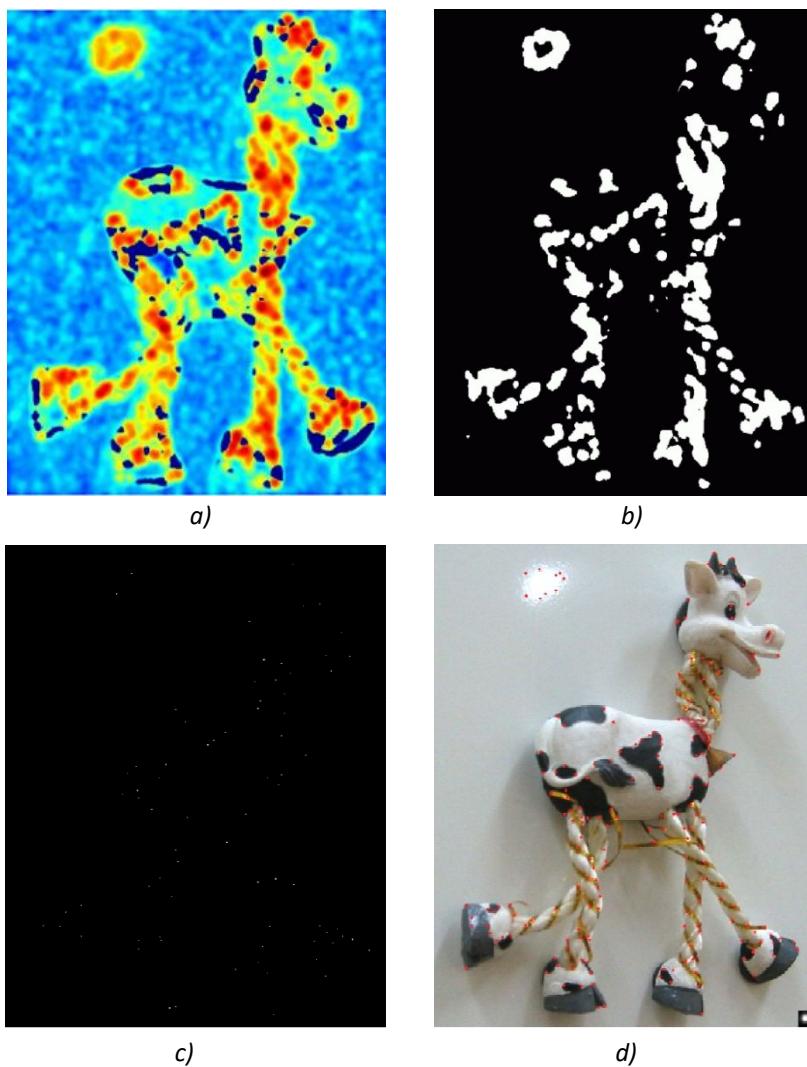
gde $w(x, y)$ predstavlja prozorsku funkciju gde se za datu funkciju najčešće usvaja pravougaona ili Gausova prozorska funkcija, dok I predstavlja pomereni i nepomerni intenzitet slike. Ova formulacija se može aproksimirati na sledeći način:

$$E(u, v) \approx [u \quad v] M \begin{bmatrix} u \\ v \end{bmatrix} \quad M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad [6]$$

gde je M 2×2 matrica izračunata iz gradijenata u slici gde je suma rađena po regionu slike koji se ispituje, I_x i I_y predstavljaju gradijente po x-osi i y-osi izračunate uz pomoć Sobelovih matrica za računanje gradijenata dok $I_x I_y$ predstavlja gradijent po x-osi pomnožen gradijentom po y-osi. Matrica M nam kaže da ako su elementi sporedne dijagonale matrice jednaki nuli, onda je u pitanju ugao koji je paralelan sa osama slike, što znači da je dominantan smer kretanja po jednoj od dve zadate ose. Ako su elementi glavne dijagonale znatno mali, znači da u tom segmentu nema ugla i da treba nastaviti dalje sa pretraživanjem drugog segmenta, zato što se matrica M može posmatrati kao elipsa sa poluprečnicima i orientacijom koji su određeni sopstvenim vrednostima i sopstvenim vektorima. Ako je jedan element glavne dijagonale mnogo veći od drugog elementa glavne dijagonale, znači da je u pitanju ivica jedne odgovarajuće ose, stoga nam je od interesa da pronađemo one segmente gde su oba elementa glavne dijagonale velika ili srazmerna jedan drugome, zato što se na tim mestima nalazi rast E u svakom pravcu što odgovara čoškovima slike. To se može dobiti sledećom formulacijom:

$$R = \det(M) - \alpha \text{trace}(M)^2 = \lambda_1 \lambda_2 - (\lambda_1 + \lambda_2)^2 \quad [7]$$

gde $\det(M)$ predstavlja determinantu matrice M, $\text{trace}(M)$ trag matrice M, α konstantu koja iznosi 0.04 - 0.06, a λ_1 i λ_2 elemente glavne dijagonale. Potrebno je naći tačke sa velikim odzivom R, odnosno poređenjem R sa nekim predefinisanim pragom i uzeti tačke u kojima R ima lokalni maksimum. To se radi tako što se uzima prozor, u ovom slučaju prozor širine 3, prolazi se kroz sliku i u svakom prozoru nalazi se pozicija maksimalnog elementa i ta pozicija se zadržava, dok se svi ostali članovi tog prozora postavljaju na nulu. Ovim postupkom dolazi do izražaja samo ona Harisova tačka koja predstavlja lokalni maksimum prozora i koja predstavlja obeležje koje će se koristiti nadalje u realizaciji rešenja. Ovaj algoritam invarijantan je na rotaciju slike, ali nije invarijantan na skaliranje, ako previše smanjimo sliku, deo slike koji je nekada predstavljao čošak sada će biti detektovan kao ivica. Implementacija ovog, kao i ostalih algoritama, biće prikazana u poglavlju Implementacija i rezultati, a na slici ispod nalazi se primer rezultata datog algoritma.



Slika 3 – Faze algoritma: a) Odziv R; b) R segmentisan pragom; c) Usvojene tačke sa lokalnim maksimumima u regionu; d) Prikaz izdvojenih obeležja.

Slike preuzete iz [4]

2.3 SIFT algoritam za ekstrakciju obeležja

SIFT algoritam (*Scale Invariant Feature Transform*) kao što mu i samo ime kaže predstavlja algoritam koji je invarijantan na skaliranje slike. Problem koji se često javlja prilikom pronađenja obeležja jeste pronaći obeležja na istim mestima na slikama koje imaju veliku razliku u skali. Kao rešenje ovog problema postavlja se zaključak da treba tražiti maksimume pogodno izabrane funkcije u određenom regionu za određenu skalu. Kada je u pitanju ovaj algoritam, on se zasniva na obrađivanju funkcije *DoG* (*Difference of Gaussian*), odnosno traženju lokalnih maksimuma njegove konvolucije sa slikom u skaliranom prostoru, pa radi odbacivanje ivica i tačaka sa malim kontrastom. Najpre se definiše prostor skale $L(x, y, \sigma)$, koji se dobije konvolucijom gausovskog kernela promenljive skale, a definisanim sledećom formulacijom:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad [8]$$

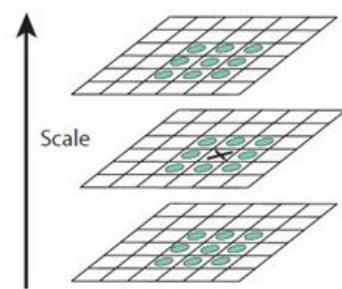
gde $G(x, y, \sigma)$ predstavlja gausovski kernel, $I(x, y)$ odgovarajuću sliku sa koordinatama x i y , σ kao parametar standardne devijacije kernela i $*$ operator zvezdice kao operator konvolucije. Gausovski kernel definisan je na sledeći način:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \quad [9]$$

Lokalni maksimumi traže se u funkciji $D(x, y, \sigma)$, koja predstavlja konvoluciju funkcije *DoG* i zadate slike, a koja se može računati iz razlike dve bliske skale prouzrokovane multiplikatorom k , čime se dobija konačna forma:

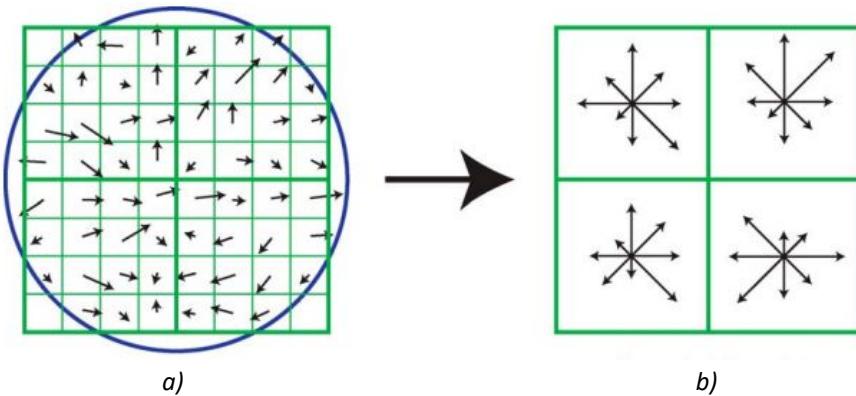
$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma) \quad [10]$$

Kako bi se detektovali lokalni maksimumi i minimumi funkcije svaka tačka upoređuje se sa 8 susednih piksela u trenutnoj slici i sa 9 suseda iz slike sledeće veće skale i prethodne skale gde se tačka proglašava za ekstremum samo ako je veća od svih ostalih tačaka ili manja od svih ostalih tačaka sa kojima se upoređuje. Nakon što su pronađeni kandidati za obeležja, potrebno je naći odgovarajuće deskriptore. Svaki posmatrani region slike podeli se na 4x4 podregione, odnosno 16 celija. Zatim se za svaki piksel unutar svakog regiona računa histogram



Slika 4 – Upoređivanje tačke sa susednim pikselima trenutne, kao i prve sledeće i prethodne skale; Slika preuzeta iz [5]

orientacija gradijenata za 8 referentnih uglova odakle se dobija da rezultujući deskriptor ima dimenzijske $4 \times 4 \times 8 = 128$. Dati pristup može se videti na sledećoj slici:



*Slika 5 – SIFT algoritam: a) Orientacija svakog piksela u 4×4 podregionima; b) Dobijeni rezultati sa 8 referentnih uglova orientacije;
Slike preuzete iz [5]*

Nakon toga vrši se eliminacija onih obeležja koje predstavljaju ivice tako što se količnik traga i determinante Hesijan matrice datog obeležja poredi sa predefinisanim pragom i određivanje konačne orientacije deskriptora. Ovaj algoritam predstavlja jednu veoma robustnu tehniku poklapanja, može da reši probleme vezane za promene u uglu slikanja sve do oko 60 stepeni rotacije ravni slike, može da reši probleme vezane za promene u osvetljaju, čak i razliku noći i dana na slici, brz je i efikasan i upotrebljiv je u realnom vremenu, zato je i danas široko rasprostranjen i predstavlja jedan od glavnih algoritama kada je u pitanju ovakva obrada.

2.4 SURF algoritam za ekstrakciju obeležja

SURF algoritam (*Speeded-Up Robust Features*) je novi algoritam koji je predložen od strane naučnika Hreberta Beja 2008. godine. Pošto ovaj operator opisuje lokalna teksturna obeležja u različitim pravcima i skalama sive slike, invarijantan je na rotaciju, skaliranje i promenu osvetljenosti, a, takođe, ima dobru stabilnost kada je u pitanju afina transformacija i pristupstvo šuma u slici, poboljšava računsku efikasnost i robustan je. *SURF* algoritam u pronalasku obeležja koristi Hesijan matricu zbog svojih dobrih performansi u pogledu potrebnog vremena izvršavanja i tačnosti. Umesto da koristi drugu meru za pronađenje lokacije i skale obeležja, Hesijan-Laplasov detektor, algoritam se oslanja na determinantu Hesijan matrice.

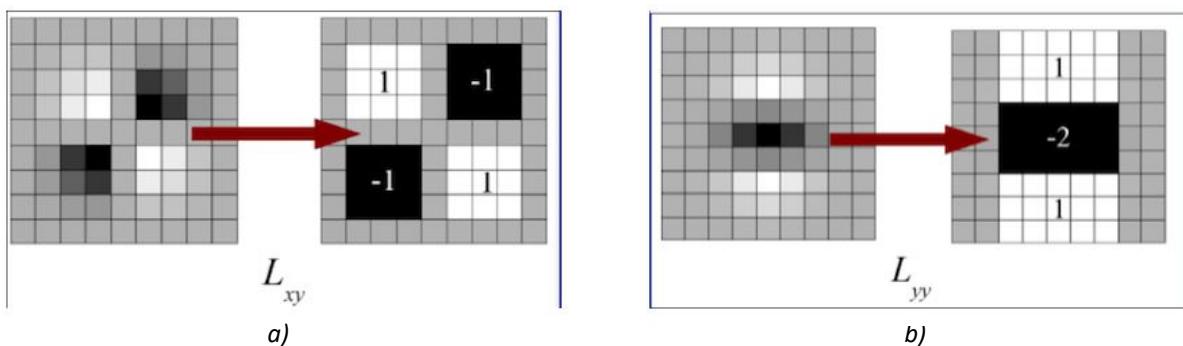
Hesijan matrica datog piksela može se definisati kao:

$$H(f(x, y)) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \quad [11]$$

gde članovi glavne dijagonale predstavljaju druge izvode po odgovarajućim osama, dok članovi sporedne dijagonale predstavljaju prvi izvod po x-osi, a zatim po y-osi. Kako bi se adaptirala na bilo koju vrednost skale, matrica se filtrira gausovskim kernelom definisanim pod relacijom [9], tako da Hesijan matrica u tački $\mathbf{X}=(x,y)$ i za skalu σ izgleda:

$$H(\mathbf{X}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{X}, \sigma) & L_{xy}(\mathbf{X}, \sigma) \\ L_{xy}(\mathbf{X}, \sigma) & L_{yy}(\mathbf{X}, \sigma) \end{bmatrix} \quad [12]$$

gde $L_{xx}(\mathbf{X}, \sigma)$ predstavlja konvoluciju gausovskog kernela i drugog izvoda po promenljivoj x, kao i slično za ostale članove. Gausovski kerneli su optimalni za analizu prostora skale, ali u praksi moraju biti diskretizovani i ograničeni. Što dovodi do gubitka ponovljivosti kod rotacije oko neparnih multiplikatora ugla $\pi/4$. Postupak se svodi prvo na množenje gausovskim kernelom, zatim na traženje drugog izvoda jer traženje gausovskih drugih izvoda zahteva mnogo manje vremena u računanju nego kod običnih drugih izvoda koristeći integralne slike i to nezavisno od veličine, zato je i SURF algoritam jako brz.



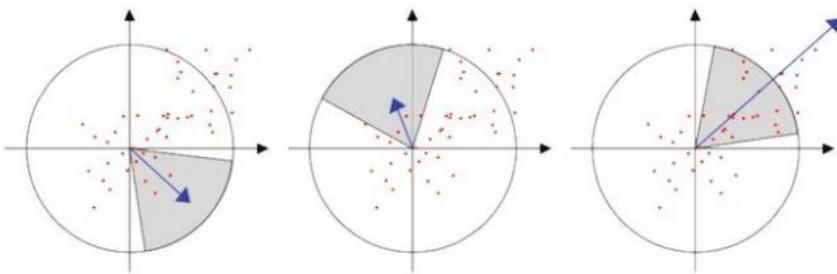
Slika 6 – a) Aproksimacija gausovskog drugog izvoda po xy; b) Aproksimacija gausovskog drugog izvoda po y;
Slike preuzete iz [6]

Ovi 9x9 filteri predstavljaju aproksimacije gausovskog drugog izvoda sa $\sigma=1.2$ i nadalje će se označavati sa D_{xx} , D_{xy} i D_{yy} . Sada se može prikazati aproksimirana determinanta Hesijan matrice kao:

$$\det(H_{approx}) = D_{xx}D_{yy} - (\omega D_{xy})^2 \quad [13]$$

gde je ω predložena konstanta jednaka 0.9.

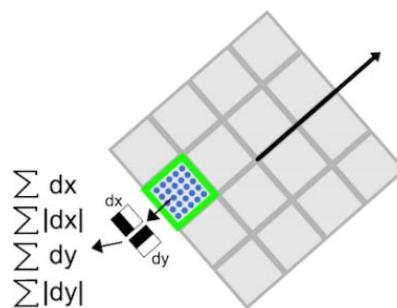
Kreiranje SURF deskriptora sastoji se iz dva koraka. Prvi korak svodi se na pronalaženje orijentacije zasnovane na kružnom regionu oko konkretnе ključne tačke, a onda se pravi kvadratni region koji je poravnat sa datom odreženom orijentacijom i iz njega se ekstrahuje SURF deskriptor. Kako bi bio invarijantan na rotaciju, algoritam pokušava da pronađe reproduktivnu orijentaciju svake pronađene ključne tačke što radi na sledeći način. Prvo se računaju odzvi na Haar talasiće u smeru osa i to u kružnom regionu sa radijusom $6s$ u okolini ključne tačke, gde s predstavlja skalu na kojoj je pronađena ključna tačka. Na visokim skalamama veličina Haar talasića je jako velika, zato se ponovo koriste integralne slike za brzo filtriranje. Zatim se računaju sume odziva na x-osi i y-osi u posmatranoj oblasti, onda se dodavanjem ugla $\pi/3$ menja orijentacija skeniranja i ponovo preračunava sve dok se ne pronađe suma sa najvećom vrednošću, što predstavlja glavnu orijentaciju obeležja.



Slika 6 – Izračunavanje glavne orijentacije obeležja preko Haar talasića

Slika preuzeta iz [6]

Sada kada je orijentacija pronađena, treba opisati obeležje. Prvo se opiše kvadratni region oko posmatrane tačke i orijentisan onom orijentacijom koja je pronađena u prethodnom koraku. Onda se region podeli u manje 4×4 podregione. Za svaki podregion računa se par prostih obeležja 5×5 . Iz razloga pojednostavljenja, sa dx i dy označavaće se wavelet odzvi na Haar talasiće. Radi robusnosti, odzvi su prvo ponderisani gausovski sa $\sigma=3.3$.



Slika 7 – Ekstrakcija komponenti deskriptora

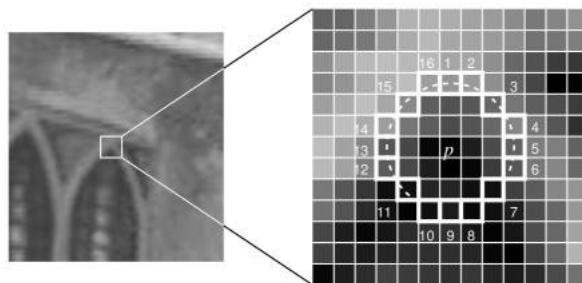
Slika preuzeta iz [6]

Onda sa wevelet odzvi dx i dy sumiraju kao što je prikazano na Slici 7 i oni predstavljaju prve osobine deskriptora. Kako bi se uvela i informacija o polaritetu intenziteta, sumiraju se i apsolutne vrednosti datih odziva. Sudeći po tome da svaki podregion ima deskriptore

dimenzijsi 4, jednakim datim sumama, a svaki podregion je dimenzija 4×4 , znači da je ukupna dužina obeležja jednaka 64, što je upola manje nego kod *SIFT* algoritma, zašto je i *SURF* algoritam dosta brži. Na nama je da odlučimo koji ćemo algoritam uzeti u zavisnosti od potražnje i uslova koje imamo.

2.5 *ORB* algoritam za ekstrakciju obeležja

ORB algoritam (*Oriented FAST and Rotated BRIEF*) je algoritam koji ima slične performanse kao *SIFT* algoritam kada je u pitanju detekcija obeležja, a brži je od njega za čak dva reda veličine. *ORB* algoritam se nadovezuje na poznati *FAST* detektor i *BRIEF* deskriptor. Obe ove tehnike su atraktivne zbog svojih dobrih performansi i niske cene. Glavni doprinosi *ORB* algoritma je dodatak brze i tačne komponente orientacije *FAST* algoritmu, efikasno izračunavanje orijentisanih *BRIEF* obeležja i analiza varijanse i korelacije orijentisanih obeležja, kao i invarijantnost na rotaciju. *FAST* je skraćenica od *Features from Accelerated and Segments Test*, što znači obeležja dobijena iz ubrzanog testa segmentacije. Algoritam funkcioniše tako što dati piksel p u slici brzo upoređuje osvetljenost sa 16 susednih piksela koji se nalaze u krugu oko datog piksela p . Zatim se pikseli u krugu sortiraju u tri klase, oni koji su svetlijii od piksela p , koji su tamniji od piksela p i oni koji su slični pikselu p . Ako je više od 8 piksela tamnije ili svetlijije od datog piksela, onda se ta tačka uzima kao ključna tačka. Tačke dobijene ovim postupkom nam daju informacije o lokaciji ivica na slici.



Slika 8 – Pronalazak ključnih tačaka *FAST* algoritmom
Slika preuzeta iz [7]

Međutim, ova obeležja nemaju komponentu orijentacije i obeležja različitih skala. *ORB* algoritam koristi piramide, odnosno više skala. Piramida slike predstavlja prikaz slike u više skala, od jako velike do jako male, koja se sastoji iz više istih slika koje se razlikuju u rezoluciji. Svaki nivo u piramidi sadrži verziju slike uzorkovanu u odnosu na prethodni nivo. Onda kada algoritam napravi piramidu, onda stupa na snagu *FAST* algoritam koji pronalazi ključne tačke na slici. Pronalaskom ključnih na svakom nivou pronalaze se obeležja za različite skale. Zbog ovog postupka može se smatrati da je *ORB* algoritam parcijalno invarijantan na skaliranje.

Nakon lociranja ključnih tačaka, dodeljuje se orijentacija svakoj tački u zavisnosti od toga kako se nivoi intenziteta menjaju oko zadate tačke. Najpre, za svaki prozor se definišu momenti:

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \quad [14]$$

Kada imamo ove momente, moguće je naći centroide, odnosno centar mase svakog prozora kao:

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \quad [15]$$

Može se konstruisati vektor od centra čoška do centroida OC , gde je njegova orijentacija jednaka:

$$\theta = \text{atan2}(m_{01}, m_{10}) \quad [16]$$

Jednom kada se izračuna orijentacija, može se rotirati u kanoničku rotaciju, a zatim izračunati deskriptor, dobijajući na nekoj vrsti invarijantnosti rotacije. Zatim *BRIEF* algoritam uzima sve pronađene tačke i pretvara ih u binarni vektor obeležja tako da zajedno mogu predstavljati objekat. Vektor binarnih obeležja poznat je i pod nazivom binarni deskriptor koji sadrži samo 1 i 0. Ukratko, svaka ključna tačka opisana je vektorom obeležja 128-512 bita. Algoritam započinje blurovanjem slike Gausovim filtrom radi prevencije deskriptora na senzitivnost visokofrekventnog šuma. Zatim se bira slučajni par susednih piksela oko posmatrane tačke. Definisano susedstvo oko piksela poznato je i kao prozor, što predstavlja kvadrat neke širine ili visine centriran oko posmatrane tačke. Prvi slučajni pixel izabran je Gausovom raspodelom centriranom oko ključne tačke i standardnom devijacijom sigma. Nakon toga, drugi pixel izabran je Gausovom raspodelom centriranom oko prvog piksela sa drugom vrednošću standardne devijacije. Ako je prvi pixel svetlij od drugog piksela, bitu se dodeljuje vrednost 1, a drugom pikselu 0 i obrnuto. Ako je u pitanju 128 bitni vektor, onda se ovaj postupak ponavlja 128 puta. Međutim, *BRIEF* algoritam nije invarijantan na rotaciju, stoga *ORB* koristi *rBRIEF* (*Rotation-aware BRIEF*) kako bi dodao ovu funkcionalnost, sa željom da se ne izgubi na brzini algoritma. Radi se binarni test $\tau(p; x, y)$ gde p predstavlja zadati prozor, a x i y pikseli, gde funkcija $\tau(p; x, y)$ dobija vrednost 1 ako je intenzitet piksela x manji od intenziteta piksela y i obrnuto ako je piksel veći ili jednak datom pikselu. Obeležje je definisano kao vektor n binarnih testova:

$$f(n) = \sum_{1 < i < n} 2^{i-1} \tau(p; x_i, y_i) \quad [17]$$

Sada možemo definisati:

$$S_\theta = R_\theta S \quad [18]$$

gde S predstavlja $2 \times N$ vektor koji odgovara odbircima x_i i y_i za svako n u prethodnom slučaju, θ predstavlja odgovarajuću orijentaciju izračunatu pre i R_θ matricu rotacije. Ovim se dobija "upravljeni" *BRIEF* algoritam i njegov operator:

$$g_n(p, \theta) = f_n(p)|(x_i, y_i) \in S_\theta \quad [19]$$

Dakle, za svaki prozor pokreće se svaki test i na samom kraju poređaju se testovi na osnovu svoje udaljenosti od srednje vrednosti formirajući vektor T . U rezultujući vektor R postavi se prvi test i obriše se iz vektora T . Gramzivom pretragom se upoređuju preostali testovi sa testovima u okviru vektora R , ako im je visok koeficijent korelacije, onda ste taj test odbacuje jer ima veliku sličnost sa nekim od već izabranih testova, u suprotnom se dodaje rezultujućem vektoru. Ovaj postupak se ponavlja sve dok se u rezultujućem vektoru ne nalazi 256 testova. Ako se desi da se u vektoru nalazi više od 256 testova, potrebno je promeniti prag tolerancije. Na samom kraju, pokazuje se da *rBRIEF* daje značajno poboljšanje kada je u pitanju varijansa i korelacija u odnosu na "upravljeni" *BRIEF* algoritam.

2.6 Algoritam za poklapanje parova obeležja

Nakon što je izvršena ekstrakcija obeležja nekom od prethodno opisanih metoda, potrebno je data obeležja adekvatno upariti, odnosno pronaći obeležja na jednoj i na drugoj slici koja predstavljaju ista ili slična obeležja koja će se nadalje koristiti u rešavanju. Postavlja se pitanje kako naći najsličnije obeležje na datim slikama. Potrebno je definisati funkciju udaljenosti koja poredi dva deskriptora i na osnovu nje određujemo da li su ta dva obeležja parovi ili nisu. Poenta je testirati sva obeležja koja pripadaju drugoj slici sa obeležjima prve slike i uzeti ona koja imaju minimalnu funkciju udaljenosti. Radi efikasnosti, u ovom slučaju, obeležja prve slike podeljena su u 32 podgrupe obeležja, pa se zatim svaka grupa obeležja prve slike upoređuje sa obeležjima druge slike. Najjednostavniji pristup jeste *SSD* pristup, odnosno *Sum of square differences*, što predstavlja sumu kvadratnih distanci između obeležja, a može se definisati kao $SSD(f_1, f_2)$, gde f_1 predstavlja obeležje prve slike, a f_2 obeležje druge slike. Međutim, mana ovog algoritma je što može imati dobre vrednosti i za pogrešne parove poklapanja, pa se zato koristi i bolji pristup koji predstavlja odnos udaljenosti, a definiše se kao:

$$\text{Odnos udaljenosti} = \frac{SSD(f_1, f_2)}{SSD(f_1, f_2')} \quad [20]$$

gde f_2 predstavlja najbolji *SSD* pronađen za poklapanje sa f_1 u slici S_2 , a f_2' predstavlja drugi najbolji *SSD* pronađen za poklapanje sa f_1 u slici S_2 . Prednost ovog algoritma je što se dobijaju male vrednosti za pogrešna poklapanja. Ovaj postupak ilustrovan je na Slici 9 na sledećoj stranici.

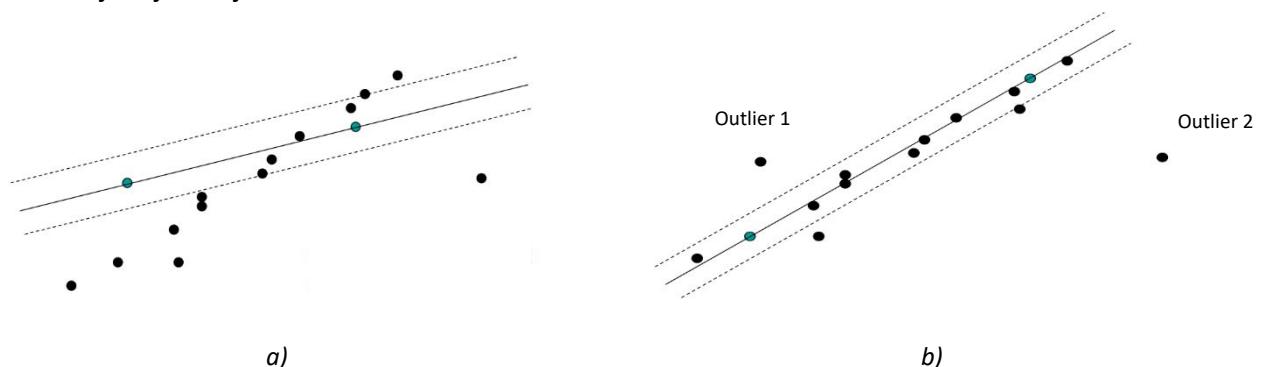


Slika 9 – a) Slika S_1 i obeležje f_1 ; b) Slika S_2 i obeležja f_2 i f_2' ;
Slike preuzete iz [7]

U opticaj za računanje razlike distanci ulaze ona obeležja koja zadovoljavaju uslov u poređenju sa predefinisanim pragom y_range koji predstavlja broj piksela odstupanja od pozicije jednog i drugog obeležja po y -osi. Zatim se kao prag za odnos udaljenosti uzima predefinisana vrednost 0.5. Ovim postupkom dolazi se do parova poklapanja obeležja koja će se koristiti u nastavku rada.

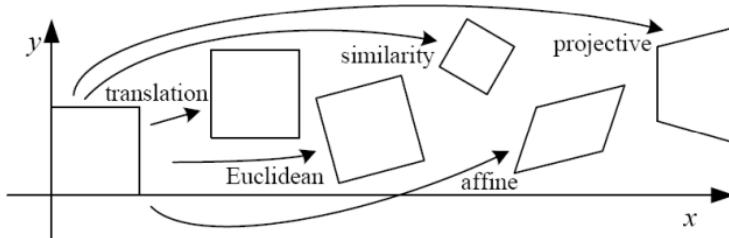
2.7 RANSAC algoritam i procena homografije

Prilikom ekstrakcije obeležja i pronalaženja parova poklapanja, može doći do pogrešnih poklapanja, odnosno loših obeležja koji kvara performanse algoritma, takva obeležja se nazivaju *outlier-ima*. Obeležja koja dobro opisuju problem i korisna su nazivaju se *inlier-i*. Kako bi se izbegao što veći broj loših obeležja, koristi se RANSAC algoritam (*Random Sample Consensus*). Cilj ovog algoritma je pronaći samo *inlier* obeležja jer ako je izabran neki *outlier* koji se koristi u izračunavanju trenutnog fita, tada preostale tačke neće imati veliki doprinos rezultujućoj krivoj.



Slika 10 – a) Prikaz lošeg fitovanja; b) Prikaz dobrog fitovanja uz pomoć RANSAC algoritma;
Slike preuzete iz [4]

Algoritam funkcioniše po sledećem principu. Ideja je nasumično izabrati set semenih tačaka na osnovu kojih će se napraviti prva procena transformacije, odnosno neka grupa ili par poklapajućih obeležja. Za svaki par poklapajućih obeležja računa se koliko su ta obeležja bliska, odnosno koje pomeranje po osama je potrebno da bi se ta obeležja poklopila, a zatim se računa razlika pomerenih obeležja druge slike i obeležja prve slike. Ako je suma kvadratnog korena kvadrirane razlike manja od zadatog predefinisanog praga, u ovom slučaju jednakom 3, dati par poklapajućih obeležja smatra se *inlier*-ima. Ako je broj *inlier*-a dovoljno veliki, ponovo izračunati procenu transformacije, ali samo na osnovu svih *inlier*-a. Kao konačna transformacija uzima se usrednjeni vektor translacije. U našem slučaju je procenjena homografija u stvari translaciona transformacija sa dve nepoznate promenljive koje predstavljaju potrebno pomeranje druge slike po x-osi i y-osi kako bi se date slike pravilno nalepile jedna na drugu. Uopšteno homografija predstavlja projektivnu transformaciju, odnosno mapiranje između bilo koje dve projektivne ravni sa istim centrum projekcije što podrazumeva da se pravougaonik mapira u neki četvorougao, da paralelizam nije obavezno sačuvan i da prave linije moraju ostati prave.



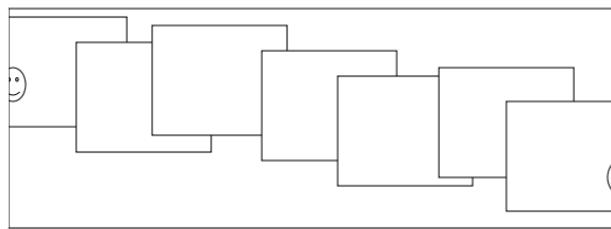
Slika 11 – Prikaz različitih projektivnih transformacija

Slika preuzeta iz [4]

Postoji više transformacija u 2D ravni kao što su translacija, rotacija, skaliranje i smicanje, a pored njih postoje i projektivne transformacije kao što su afina transformacija i projektivna uvijanja. U našem slučaju korišćena je 2D translaciona transformacija koja ima oblik:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad [21]$$

gde t_x i t_y predstavljaju adekvatna pomeranja po x-osi i y-osi respektivno. Jednom kada je pronađena translaciona matrica, potrebno je izvršiti spajanje slika. Algoritam lepljenja slika funkcioniše prostom translacijom slike duž koordinata. Prva slika skrati se za onoliko za koliko je predviđeno pomeranje druge slike uлево, a zatim se druga slika nalepi na prvu, i tako sve dok ima ulaznih slika, pri čemu je leva slika uvek prethodno spojena slika iz prošle petlje. Ovim postupkom dobijaju se panorame. Pri ovom postupku moguće je da dođe do neželjenih efekata kao što je zanošenje (skretanje) slika.



Slika 12 – Problem zanošenja slika

Slika preuzeta iz [4]

Male vertikalne greške se postepeno akumuliraju i mogu dovesti do problema prikazanog na Slici 12. Jedan od načina rešavanja ovog problema biće prikazan u sledećem poglavlju.

2.8 Algoritmi *blending-a*, poravnavanja i isecanja panorame

Pored toga što prilikom spajanja slika može doći do zanošenja (skretanja) slika, na spojevima slika se javlja efekat poruba, odnosno u većini slučajeva jasno se na panorami nazire mesto na kojem su date slike spojene, što je razlog različite osvetljenosti slika u datim trenucima. Kako bi dobijena panorama izgledala što uverljivije, potrebno je ukloniti te tragove nastale spajanjem slika. Algoritam koji se time bavi naziva se *Image blending*. Algoritam radi tako što posmatra samo tačke koje se nalaze blizu spoja. Ako se tačka nalazi daleko od spoja, onda se ona preskače. Obično se kao merilo blizine poruba uzima prozor, u ovom slučaju dimenzija 20x20 zato što su u pitanju slike sa malo većom rezolucijom. Kada se dođe do tačke koja se nalazi dovoljno blizu poruba slike, uzimaju se vrednosti intenziteta piksela levo i desno od poruba i sledećom relacijom dolazi se do novog intenziteta slike:

$$ratio = (x_{tren} - porub_{index} + dim_{prozor}) / (2 * dim_{prozor}) \quad [22]$$

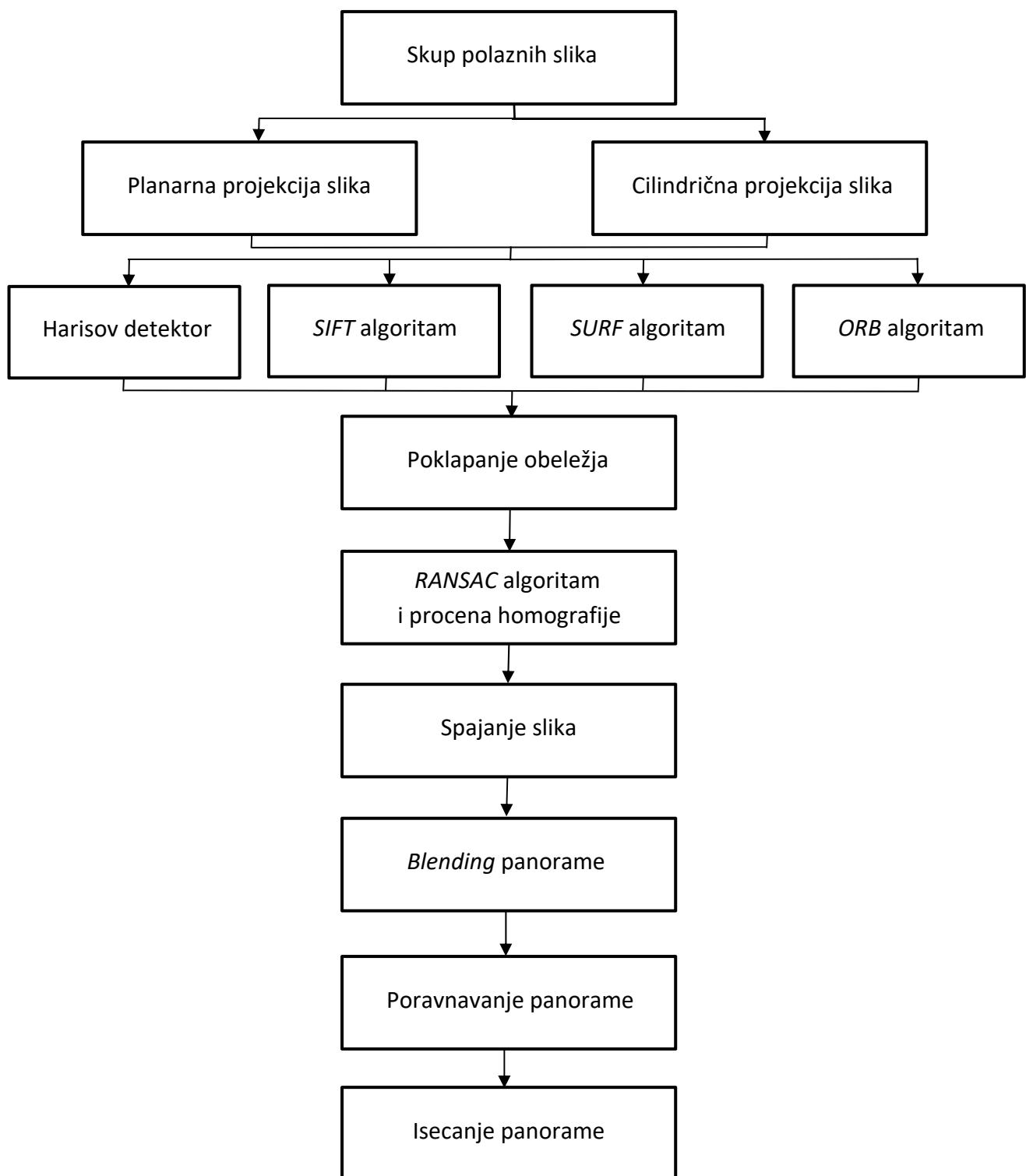
$$x_{tren. novo} = (1 - ratio) * intenzitet_{levi piksel} + ratio * intenzitet_{desni piksel} \quad [23]$$

Ovim postupkom se vrši neki vid usrednjavanja čime se dobija uglačana verzija prelaza između dve slike i smanjuje se efekat poruba. Vratimo se na problem zanošenja slike sada. Rešavanje ovog problema u praksi naziva se *Image Straightening*, a zasniva se na ispravljanju akumulirane greške zasnovane na vertikalnom pomeranju. Najpoznatiji i najteži metod rešavanja ovog problema zove se *Bundle Adjustment*, pored toga postoji i način sa dodavanjem početne slike na kraj panorame, čime se ovaj problem smanjuje, ali ovim metodima se nećemo baviti u radu. Rešenje problema bazirano je na sledećem postupku. Najpre se saberi sva pomeranja po x-osi i y-osi i odredi apsolutna vrednost sume po y-osi. U zavisnosti od toga da li je znak obe sume isti ili nije, možemo zaključiti da li je došlo do

zanošenja slike nadole ili nagore. Na osnovu toga, može se generisati niz ravnomerno raspoređenih tačaka od 0 do absolutne vrednosti pomeranja po y-osi sa ekvivalentnim razmacima i to onoliko tačaka kolika je ukupna širina slike. Zatim idući redom kroz sliku sa leva na desno vrši se poravnavanje slike određenom generisanim tačkom čime se efekat zanošenja slike uklanja. Na samom kraju, kada je sve gore navedeno ispoštovano, dobije se kao rezultat panorama sa crnim okvirom, odnosno viškom slike. Ova pojava rešava se prostim odsecanjem slike sa svih strana do otklanjanja crnog okvira.

3 METODOLOGIJA RADA

Kao što je već poznato, cilj ovog rada je uspešna implementacija algoritma za spajanje slika u panorame različitih projekcija sa više vrsta obeležja. Na raspolaganju je 30 fotografija lično sakupljenih od strane autora. Slike su tako koncipirane da obuhvataju više segmenata. Podeljene su u četiri direktorijuma i to tako da se u njima nalaze slike enterijera sa istaknutim kućnim elementima i nameštajem, slike koje većim delom obuhvataju urbani deo, sa zgradama, kućama i elementima zelenih površina, slike koje pretežno obuhvataju prirodu i slike korišćene za izradu panorame u 360 stepeni. Napravljen je ovakav koncept iz razloga da se pokaže kako algoritam radi za različite tipove slika i problema. Slike su prikupljene uz pomoć Nikon DSLR D5200 fotoaparata sa objektivom Nikkor 18-140mm f/3.5-5.6G ED VR koristeći minimalan otvor blende, odnosno otvor koji odgovara žižnoj daljini od 18mm. Slike su prikupljene u portret format dimenzija 6000x4000, pa su radi efikasnosti algoritma korišćene smanjene rezolucije dimenzija 1200x800. Pri pristupanju ovom problemu potrebno je uzeti u obzir sve prethodno navedene teorijske osnove i algoritme koji su opisani i to ukombinovati u jednu kompaktnu celinu koja će uspešno rešavati zadati problem. Implementacija ovog algoritma rađena je u programskom jeziku *Python*, u verziji 3.7. Takođe, pored samog programskog jezika, potrebno je imati instalirane i određene biblioteke kao što su *OpenCV*, odnosno *cv2*, *os*, *numpy*, *math* i *matplotlib*. Biblioteka *OpenCV* sadrži funkcije i potrebne algoritme kompjuterske vizije i digitalne obrade slike, *os* biblioteka sadrži osnovne funkcije za učitavanje fajlova, *numpy* biblioteka sadrži funkcije za rad sa nizovima i matricama, *math* biblioteka sadrži matematičke funkcije i biblioteka *matplotlib* služi za iscrtavanje grafika, slika i potrebnih rezultata. Samo rešavanje problema sastoji se iz više tačaka. U zavisnosti od odabranih podalgoritama, postupak rešavanja je malo drugačiji i dobijaju se drugačiji rezultati. Najpre se posmatraju ulazni podaci koji predstavljaju prikupljene slike ranije navedenih tipova. U zavisnosti od odabira željene projekcije, u nastavku algoritma koriste se slike u planarnoj ili u cilindričnoj projekciji. Na tako dobijenim slikama potrebno je izvršiti ekstrakciju obeležja, odnosno naći odgovarajuće deskriptore na slikama koji će se koristiti u daljem radu, a tako da ta obeležja budu što informativnija. U zavisnosti od odabira ponuđenih algoritama, obeležja se mogu izdvojiti na 4 različita načina: pomoću Harisovog detektora, odnosno pronalaženja Harisovih tačaka koje detektuju čoškove na slikama, pomoću *SIFT* algoritma, *SURF* algoritma ili *ORB* algoritma. Na osnovu izabranog algoritma, dobijaju se različita obeležja nakon čega se vrši prikaz istih radi uvida u sam rezultat datog algoritma. Nakon toga, izdvojena obeležja je potrebno dalje tretirati kako bi bila spremna za algoritam spajanja slika. Na sledećoj stranici na Slici 13 nalazi se blok dijagram implementacije celokupnog algoritma o čijim će detaljima biti reči u nastavku poglavila.

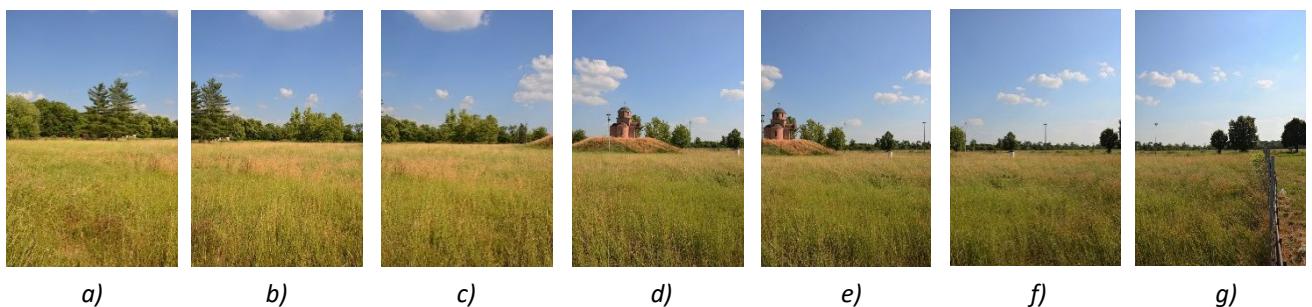


Slika 13 – Blok dijagram rešavanja

Kao što je rečeno, potrebno je date algoritme opisane u prethodnom poglavlju ukombinovati u jednu kompaktnu celinu. Predlog konstrukcije algoritma dat je na prethodnoj slici blok dijagramom. Celokupna metodologija sastoji se od sledećih koraka. Kao što je već opisano, prvo se iz ulaznih slika vrši ekstrakcija obeležja. U zavisnosti od izabranog algoritma, dobijaju se različita obeležja jer svaki od datih algoritama radi na različit način. Nakon toga vrši se prikaz dobijenih obeležja gde su na svakoj pojedinačnoj slici označena pronađena obeležja. Potom je potrebno pronaći parove poklapanja obeležja za svake dve slike između kojih se vrši spajanje, odnosno pronaći ona obeležja na obe slike koja su najsličnija prethodno opisanim algoritmom. Cilj je testirati sva obeležja druge slike sa obeležjima prve slike i pronaži onaj par za svako obeležje koji daje minimalnu funkciju udaljenosti, a tu funkciju u ovom slučaju predstavlja odnos udaljenosti gde se kao prag udaljenosti uzima predefinisana vrednost 0.5. Nakon što smo pronašli navodne parove poklapanja obeležja, na red dolazi *RANSAC* algoritam. Ovaj algoritam služi za odbacivanje *outlier-a*, što u stvari predstavlja loša obeležja koja kvare performanse algoritma. Za svaki par poklapajućih obeležja računa se koliko su ta obeležja bliska, odnosno koje pomeranje po osama je potrebno da bi se ta obeležja poklopila, a zatim se računa razlika pomerenih obeležja druge slike i obeležja prve slike. Ako je razlika manja od zadatog predefinisanog praga, dati par poklapajućih obeležja smatra se *inlier-ima*. Kao konačna transformacija uzima se usrednjeni vektor translacije. U našem slučaju je procenjena homografija u stvari translaciona transformacija sa dve nepoznate promenljive koje predstavljaju potrebno pomeranje druge slike po x-osi i y-osi kako bi se date slike pravilno nalepile jedna na drugu. Nakon ovoga, potrebno je dobijeno pomeranje iskoristiti za spajanje slika. Proces se obavlja tako što se druga slika nalepi na prvu sliku tačno na onom mestu koje je određeno translatornim pomeranjem po osama. Ovaj postupak se ponavlja sve dok ima slika za spajanje. Prilikom ovog spajanja može doći do neželjenih pojava kao što je skretanje, odnosno zanošenje slike. Ovaj problem rešen je uz pomoć ukupnog vektora pomeranja po osama svih slika, gde se računanjem ukupnog pomeranja po obe ose i konstruisanjem vektora ekvidistantnih tačaka svaka od kolona ispravlja odgovarajućim brojem, čime se dobija ispravljena slika. Drugi problem koji može nastati je uočljiva pojava poruba na mestima gde se nalazi spojevi slika zbog razlika u osvetljenosti i intenzitetu slika na datim mestima. Ovaj problem rešava se uz pomoć algoritma za *blending* slike, opisanog u prethodnom poglavlju, čime se praktično vrši neka vrsta usrednjavanja na samim porubama čime se okolnim pikselima dodeljuju slične vrednosti kako bi se sam prelaz manje video. Na samom kraju, kada su svi prethodni koraci odrađeni, dobijena je panorama sa crnim okvirom, odnosno viškom, pa je potrebno taj višak ukloniti. Odsecanjem crnog okvira sa sve četiri strane algoritmom isecanja dobija se konačna panorama. Nešto više o samoj implementaciji ovih algoritama i detaljnom opisu kompletног postupka biće priče u sledećem poglavlju koje nosi naziv **Implementacija i rezultati**.

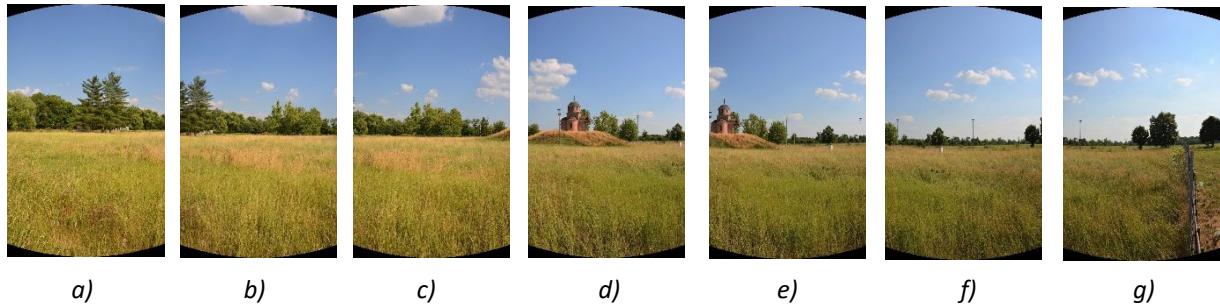
4 IMPLEMENTACIJA I REZULTATI

U ovom delu biće više reči o samoj implementaciji algoritma, njegovim detaljima i dobijenim rezultatima posmatranim na egzaktnim primerima, odnosno slikama prikupljenim od strane autora. Najpre je potrebno učitati sve navedene potrebne biblioteke koje se koriste u programskom jeziku *Python*. Na samom početku definišu se konstante koje se koriste u implementaciji kao što su: *flag cylinder* koji ima vrednost 'true' ili 'false' i govorи nam о tome da li želimo da spajamo slike u planarnoj ili cilindričnoj projekciji, *algoritam* kojom odabiramo koji ćemo od 4 moguća algoritma iskoristiti za ekstrakciju obeležja i može imati vrednosti 'Harris', 'SIFT', 'SURF' ili 'ORB', *plot* kojom odlučujemo da li želimo da program izbacuje grafike i rezultate svih međukoraka koja može imati vrednosti 'true' ili 'false', *feature_threshold=0.01* kojom se odlučuje o prihvatanju obeležja ili ne u Harisovom algoritmu, *descriptor_size=5* koja određuje dimenzije obeležja, *ransac_k=1000* kojom se definiše veličina semene grupe u *RANSAC* algoritmu, *ransac_threshold_distance=3* kojom se definiše prag distance u *RANSAC* algoritmu i *alpha_blend_window* kojom se definiše dimenzija prozora za *blending* slike. Date definisane vrednosti korišćene su u algoritmu, iako same vrednosti mogu varirati od vrste problema i mogu se naknadno podešavati. Najpre se učitavaju slike iz željenog foldera koji će se obrađivati i žižne daljine koje odgovaraju tim slikama u slučaju da želimo da radimo sa slikama u cilindričnoj projekciji. Kako rad ne bi bio prenatrpan prikazivanjem slika u svakom koraku za svaki folder, detaljno će biti opisan algoritam na primeru jednog fodlera slika, dok će se za ostale slike prikazati samo dobijeni rezultati. Zatim u zavisnosti od *flag-a cylinder* pozivamo funkciju *cylindrical_system()* kojom se vrši konverzija slika iz planarne projekcije u cilindričnu. Nakon toga, u zavisnosti od odabranog algoritma za ekstrakciju obeležja ulazimo u odgovarajuće petlje programa. Kako je moguće odabrati jedan od 4 algoritma, u nastavku će biti prikazani rezultati u sva 4 slučaja. Najpre se može videti izgled ulaznih slika:



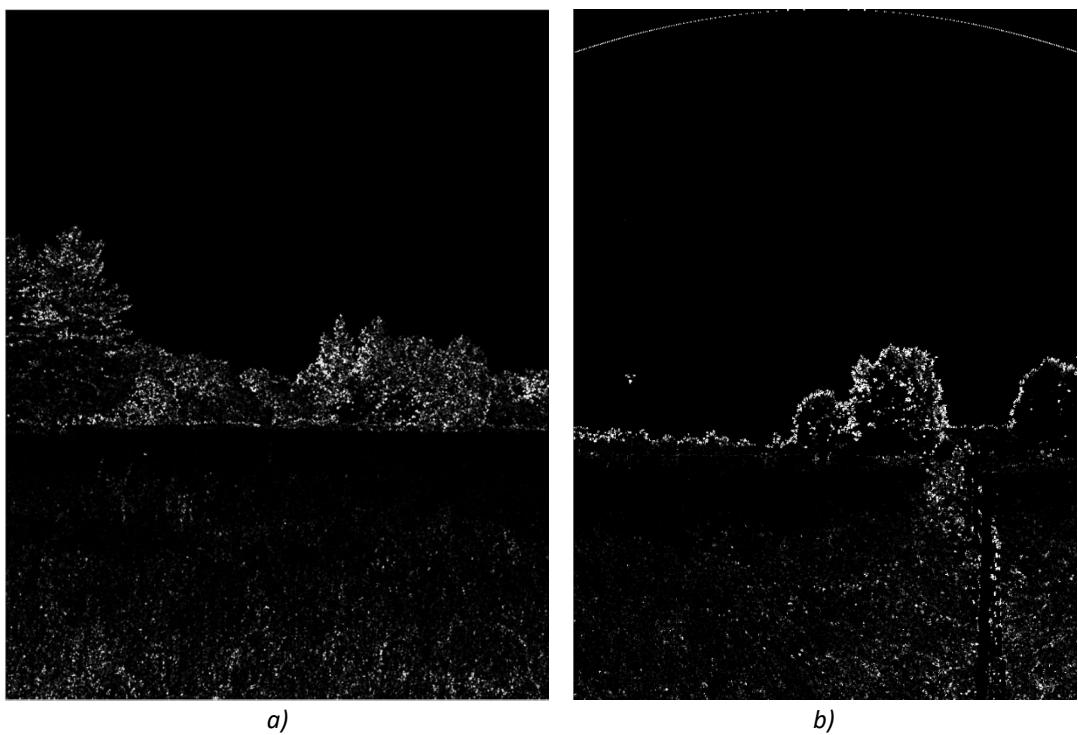
Slika 14 – a-g: Ulazne slike u planarnoj projekciji

U slučaju da su slike u cilindričnoj projekciji, tada one dobijaju sledeći izgled:



Slika 15 – a-g: Ulazne slike u cilindričnoj projekciji

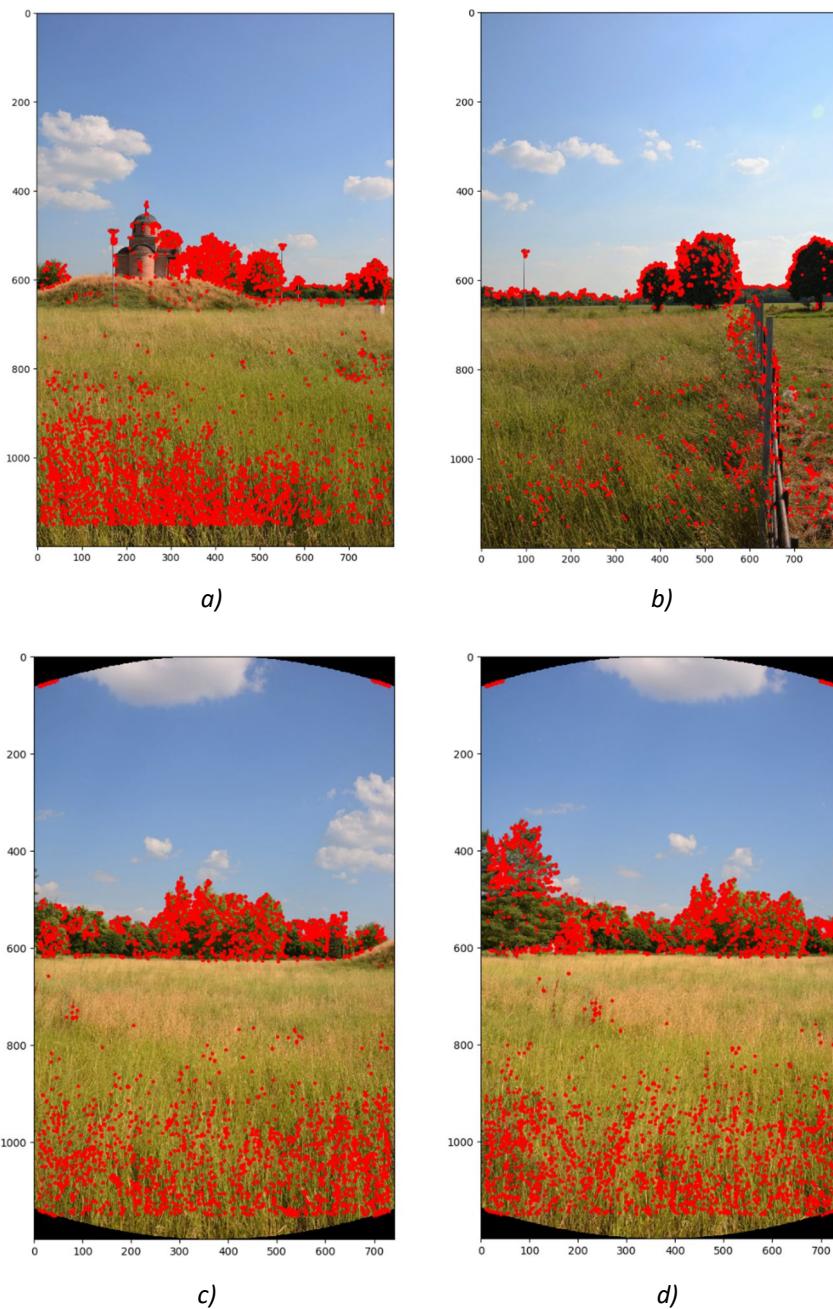
Nakon učitavanja slika i prebacivanja u cilindričnu projekciju, potrebno je izdvojiti obeležja. Prvi posmatrani algoritam biće Harisov detektor čoškova. Pozivanjem funkcije *harris_dots()* koja u okviru sebe poziva funkciju za računanje pomenute R matrice *R_parameter()* po relaciji (7) dobijaju se tačke koje predstavljaju čoškove na slikama. Na Slici 16 biće prikazani rezultati ove funkcije. Kako su rezultati za svaku sliku manje-više slični, biće prikazane po jedna slika za planarnu i po jedna slika za cilindričnu projekciju.



Slika 16 – a) Prikaz Harisovih tačaka druge u nizu slike planarne projekcije; b) Prikaz Harisovih tačaka sedme u nizu slike cilindrične projekcije

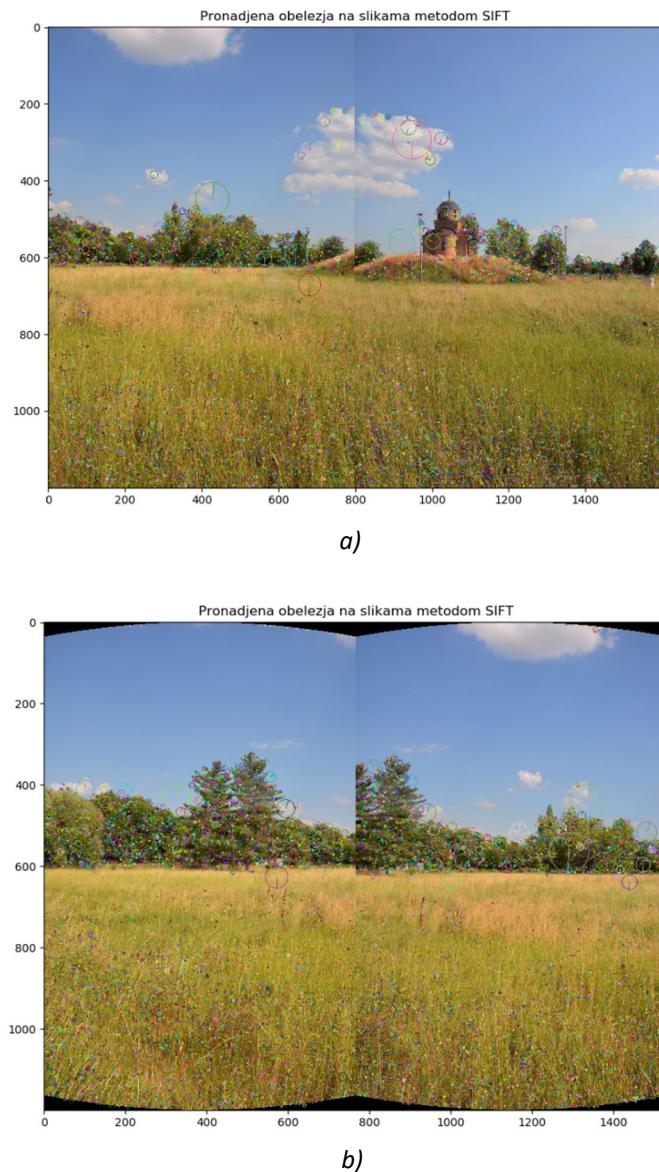
Sa datih slika moguće je uočiti da je Harisov detektor uspešno izdvojio tačke čoškova jer već golim okom možemo nazreti sličnost između originalnih i dobijenih slika. Takođe, na Slici 16 b) može se uočiti artefakt cilindrične projekcije, kako je algoritam izdvojio i tačke ivica slike.

U nastavku, potrebno je od ovih tačaka izdvojiti adekvatna obeležja. Obeležja se izdvajaju opisanom metodom lokalnih maksimuma u zadatom regionu, pozivanjem funkcije `extract_features()` sa odgovarajućim slikama, prethodno definisanim dimenzijama obeležja i pragom. Kao rezultat funkcije dobijaju se obeležja, u ovom slučaju opisana sa 5 brojeva, čime ukupna dimenzija deskriptora iznosi $broj_obeležja \times 5$, kao i pozicije ključnih tačaka tih obeležja. Na Slici 17 biće prikazani neki od rezultata dobijeni nakon ovog koraka:



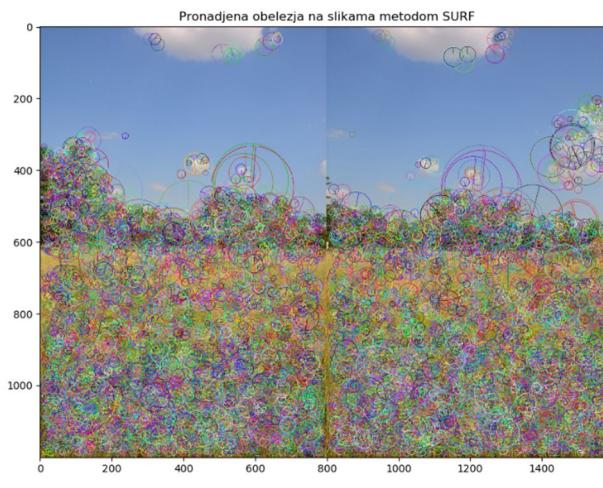
Slika 17 – a) i b): Dobijena obeležja na slikama planarne projekcije; c) i d): Dobijena obeležja na slikama cilindrične projekcije

Ovako dobijena obeležja koriste se dalje u rešavanju algoritma. Problem ovog algoritma je što nije invarijantan na skaliranje, zato se koriste i drugi algoritmi. Prvi od tih algoritama jeste *SIFT* algoritam. Ovaj algoritam predstavlja bolje rešenje od Harisovog detektora jer je invarijantan na skaliranje i rotaciju, ali je zato računski zahtevniji. Za pristup ovom algoritmu potrebno je imati instaliranu biblioteku *OpenCV*. Najpre se napravi detektor pozivanjem funkcije *cv2.xfeatures2d.SIFT_create()*, pa se zatim dati deskriptori i ključne tačke dobijaju pozivanjem funkcije datog detektora, odnosno njegove funkcije *detectAndCompute()* prosleđujući joj kao argument sivu sliku. Ovako dobijena obeležja prikazana su na slici ispod. Obeležja su kružnog oblika opisujući maksimalnu moguću površinu kojom se dato obeležje može opisati, zajedno sa orientisanim poluprečnikom u smeru orientacije izračunate u prethodno objašnjrenom algoritmu čime se dobijaju dimenzije obeležja *broj_obeležja x 128*.

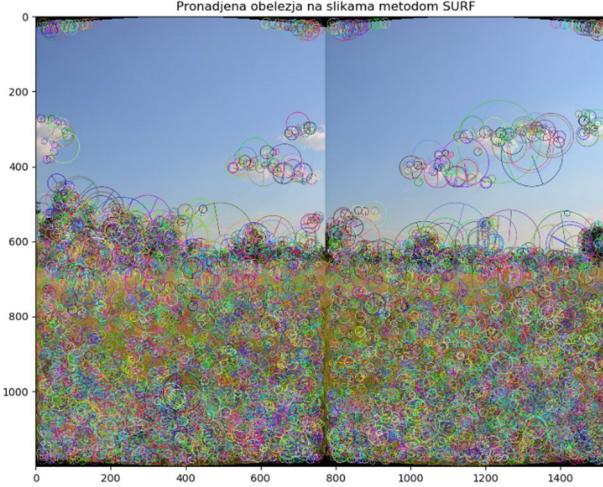


Slika 18 – a) Pronađena obeležja na dvema slikama planarne projekcije; b) Pronađena obeležja na dvema slikama cilindrične projekcije

Ovim algoritmom izdvojena su obeležja dimenzija jendakim $broj_obeležja \times 64$. Ovo je čak upola manje od *SIFT* algoritma, zašto je i ovaj algoritam dosta brži. Takođe, invarijantan je na skaliranje, rotaciju i promenu osvetljenja i robustan je. Čak bi se moglo kombinovati i naizmenične slike noći i dana i algoritam bi i dalje radio dobro, kao i prethodni. Za ovaj algoritam, potrebno je, takođe, imati biblioteku *OpenCV*. Pozivanjem funkcije `cv2.xfeatures2d.SURF_create()` pravi se detektor *SURF* obeležja, čijom se funkcijom `detectAndCompute()` dobijaju tražena obeležja i ključne tačke njihovih pozicija. Na Slici 19 mogu se videti dobijena obeležja na primerima slika planarne i cilindrične projekcije.



a)

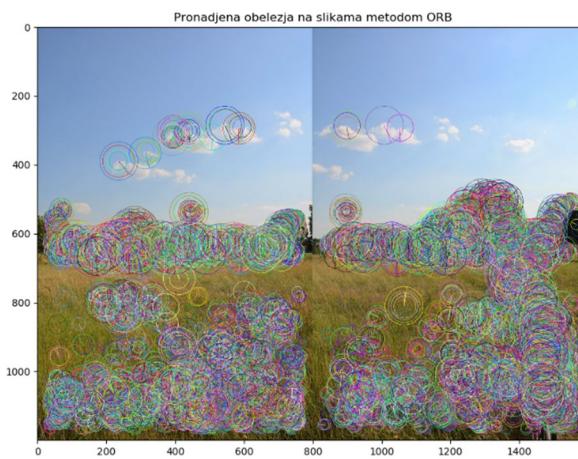


b)

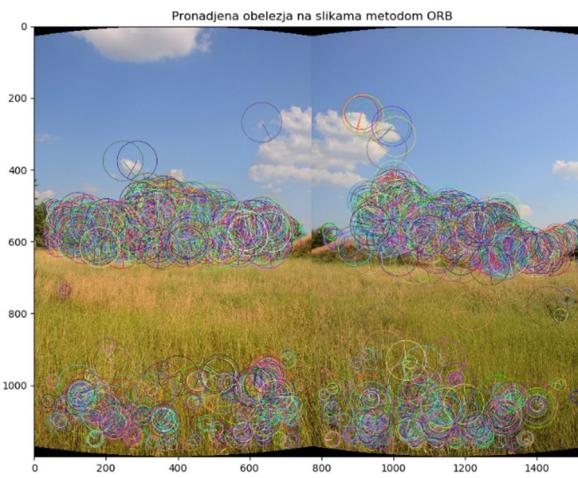
Slika 19 – a) Pronadjeni obeležja na dvema slikama planarne projekcije; b) Pronadjeni obeležja na dvema slikama cilindrične projekcije

Na datim slikama može se primetiti da ima više obeležja nego u prethodnom algoritmu i to većih dimenzija, kao i pojave pronalaska loših obeležja na čoškovima cilindričnih slika.

Poslednji algoritam kojim se može izvršiti ekstrakcija obeležja je *ORB* algoritam. Ima iste performanse kao *SIFT* algoritam, a brži je od njega čak za dva reda veličine. Ovaj algoritam se nadovezuje na poznati *FAST* detektor i *BRIEF* deskriptor. Obe ove tehnike su atraktivne zbog svojih dobrih performansi i niske cene. Iz razloga što ovaj algoritam pronađi veliki broj obeležja, broj obeležja u ovom slučaju ograničen je na 3000. Inače, kako u ovom, tako i u prethodnim algoritmima, broj obeležja se meri u hiljadama, a od svih tih obeležja nalazi se oko 100 adekvatnih parova poklapanja, o čemu će biti reči kasnije. Za ovaj algoritam potrebno je imati biblioteku OpenCV i pozivanjem funkcije `cv2.ORB_create(nfeatures=3000)` kreira se detektor čijom se funkcijom `detectAndCompute()` dobijaju obeležja i ključne tačke. Prikaz dobijenih obeležja, kako u ovom, tako i u prethodna dva algoritma omogućen je uz funkciju `drawKeypoints()`. Na Slici 20 nalaze se dobijena obeležja ovim algoritmom.



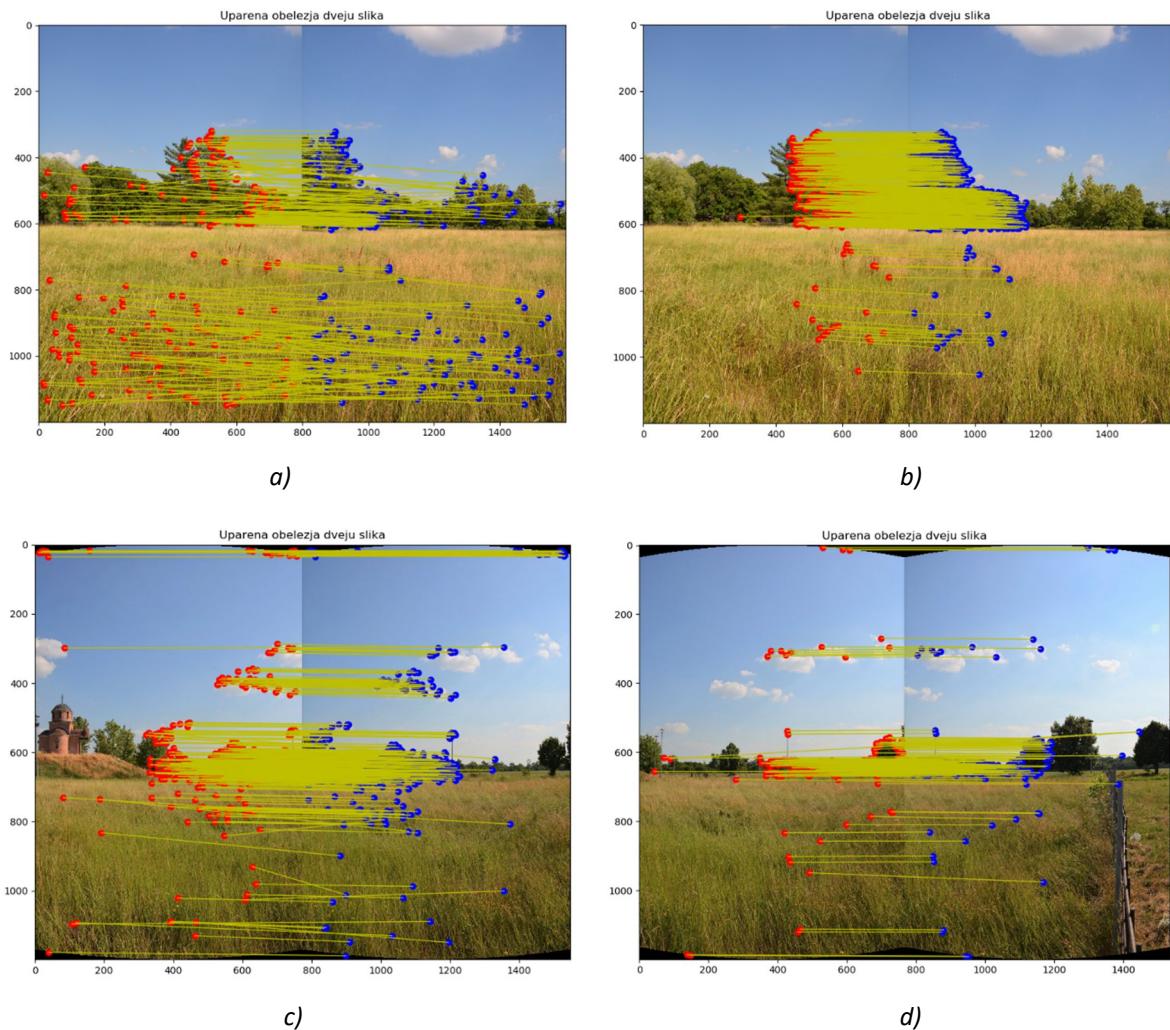
a)



b)

Slika 20 – a) Pronadjeni obeležja na dvema slikama planarne projekcije; b) Pronadjeni obeležja na dvema slikama cilindrične projekcije

Na Slici 20 moguće je uočiti da su ova obeležja daleko većeg poluprečnika nego obeležja prethodna dva algoritma. Sama činjenica da se slikom prelazi prozorima većih dimenzija i tražeći veća obeležja, za to je potrebno mnogo manje vremena nego za opisivanje obeležja jako malih dimenzija. Takođe, može se primetiti da nema pojave pronalaska obeležja u čoškovima slika, kao što je to bio slučaj sa prethodnim algoritmom. Nakon pronalaska obeležja, potrebno je naći parove poklapanja na slikama, odnosno one parove koji predstavljaju ista ili jako slična obeležja na slikama. To se može ostvariti pozivom funkcije *matching_features()* kojoj se prosleđuju prethodno dobijena obeležja i njihove pozicije gde se poklapanje računa na osnovu odnosa udaljenosti. Iz razloga što se u nastavku rada preostali algoritmi ne menjaju previše, odnosno za sva dobijena obeležja algoritmi rade skoro identičnu stvar, biće prikazivani primeri za svaku vrstu obeležja u planarnoj ili cilindričnoj projekciji. Tako se pronađeni parovi poklapanja mogu videti na Slici 21.



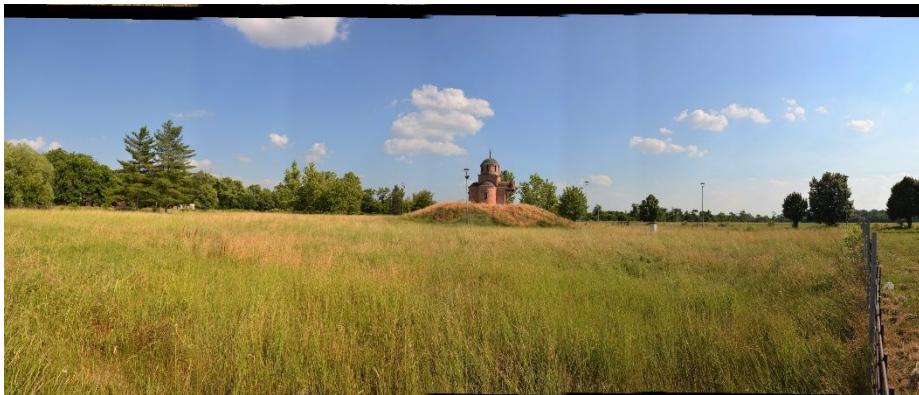
Slika 21 – a) Parovi poklapanja slika planarne projekcije obeležja Harisovog detektora; b) Parovi poklapanja slika planarne projekcije obeležja SIFT detektora; c) Parovi poklapanja slika cilindrične projekcije obeležja SURF detektora; d) Parovi poklapanja slika cilindrične projekcije obeležja ORB detektora

Na Slici 21 mogu se videti parovi poklapanja na slikama. Takođe, može se uočiti da ima i dobrih parova i onih loše pronađenih parova koji utiču na performanse algoritma. Potrebno je pronaći samo ona dobra poklapanja, a loša poklapanja zanemariti. Ovaj postupak ispunjava se pozivanjem funkcije *RANSAC()* koja pronalazi najsličnija obeležja i traži najbolje pomeranje slike kako bi se izvorna i odredišna slika što bolje uklopile, gde najbolje pomeranje predstavlja ono koje ima najviše *inlier* obeležja. Dato pomeranje u stvari predstavlja translacionu transformaciju slike. Koristeći dobijeno najbolje pomeranje vrši se spajanje slika, tako što se na mestu spoja dve slike, izvorna slika odseče, a zatim se na nju nalepi nova slika, a pozivanjem funkcije *stitching_images()*. Pre nego dođe na red upoređivanje rezultata svih algoritama, na jednom primeru rezultata biće ilutrovani preostali algoritmi. Spajanjem slika dobija se sledeća slika:



Slika 22 – Panorama dobijena nakon spajanja svih slika

Na Slici 22 može se videti dobijena panorama nakon spajanja svih slika. Kada bi se ova slika pokazala nekome ko nije iz ovog sveta, kako bi se reklo, odgovor bi bio da je ovo jedna obična panorama, ali ako se malo bolje zagleda u sliku, mogu se pronaći mesta spojeva slika. Ovaj efekat je u mnogome smanjen korišćenjem *blending*-a slike, koji je implementiran u okviru same funkcije za spajanje slika. Ovaj problem nastaje zbog razlika u intenzitetima piksela na porubima spojenih slika, pa se na neki način usrednjavanjem piksela na porubima i okolnih piksela taj efekat znatno smanjuje, a može se i potpuno ukloniti, kao što je to slučaj na ovoj panorami kada su u pitanju prva dva poklapanja slika, dok se u kasnijim poklapanjima ovaj problem ipak i dalje može uočiti. Sudeći po tome da je ovo ručna implementacija algoritma, ovo su skroz zadovoljavajući rezultati. Drugi problem koji se može javiti prilikom spajanja slika jeste skretanje(zanošenje) slike nadole ili nagore. Ovaj problem nastaje kada se vremenom akumilira greška koja potiče od vertikalne komponente spajanja. Na Slici 22 možemo uočiti efekat skretanja nagore. Ovaj problem rešava se poravnavanjem slike, što se još i naziva *Image Straightening*. Poravnavanje panorama izvršava se pozivanjem funkcije *align_image()*. Dobijene slike obe projekcije nakon ove obrade prikazane su na Slici 23 i Slici 24.

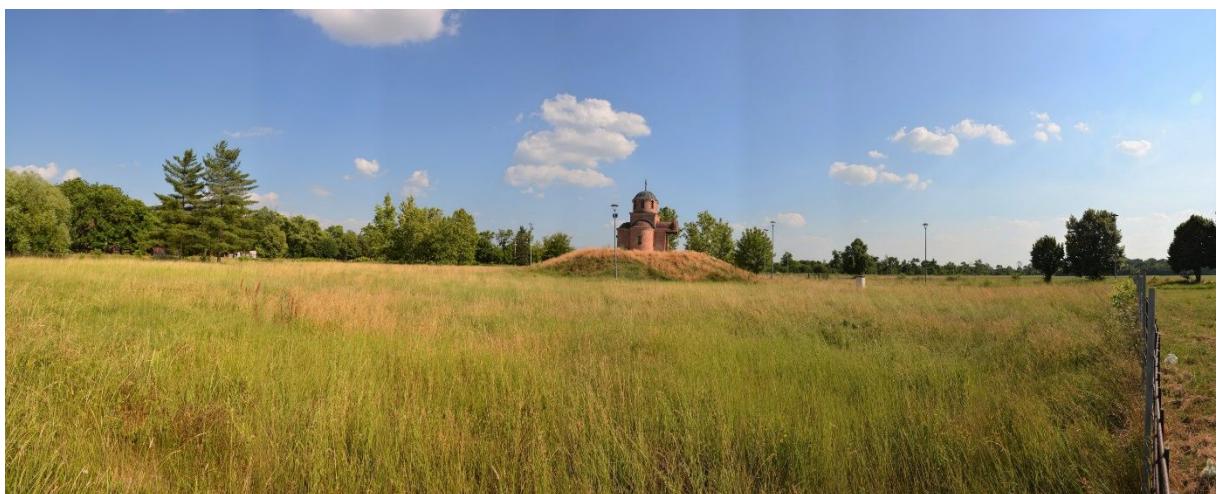


Slika 23 – Poravnata panorama planarne projekcije nakon Image Straightening-a



Slika 24 – Poravnata panorama cilindrične projekcije nakon Image Straightening-a

Na Slici 23 i Slici 24 može se videti otklonjeni efekat skretanja slike. Na samom kraju potrebno je iseci crni okvir viška oko slike. Ovaj postupak izvršava se pozivanjem funkcije *cropping()*. Konačna dobijena panorama prikazana je na Slici 24.



Slika 25 – Konačna dobijena panorama

Na Slici 25 može se videti panorama koja se dobija na samom kraju algoritma. Ovim se potvrđuje uspešnost algoritma kada je u pitanju spajanje slika. Međutim, kao i u svakom realnom problemu postoje drugačiji primeri i algoritam treba da radi dobro za svaki primer. U nastavku poglavlja akcenat će biti na upoređivanju dobijenih panorama koristeći sva 4 predložena algoritma za izdvajanje obeležja, kako planarnih, tako i cilindričnih projekcija slika kao i korišćenjem ugrađene funkcije biblioteke *OpenCV* koja sve ove navede korake radi sama. Pored slike naselja, uz koju je opisan celokupni algoritam, biće prikazani i rezultati slika iz preostala dva direktorijuma.

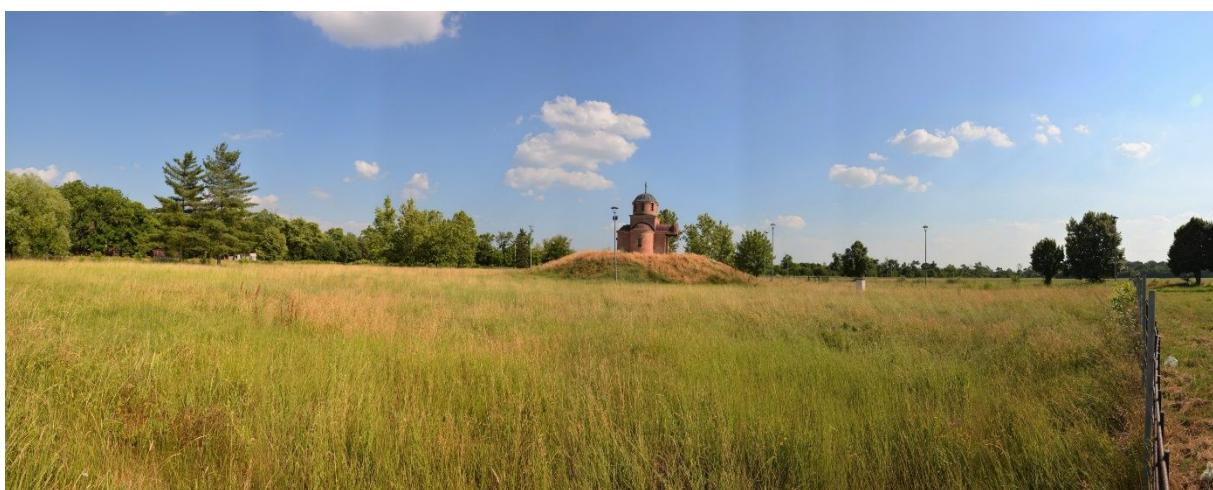
1. primer: Rezultati korišćenih algoritama-planarna projekcija

Harisov algoritam



Slika 26 – Konačna dobijena panorama Harisovim algoritmom za ekstrakciju obeležja

SIFT algoritam



Slika 27 – Konačna dobijena panorama SIFT algoritmom za ekstrakciju obeležja

SURF algoritam



Slika 28 – Konačna dobijena panorama SURF algoritmom za ekstrakciju obeležja

ORB algoritam

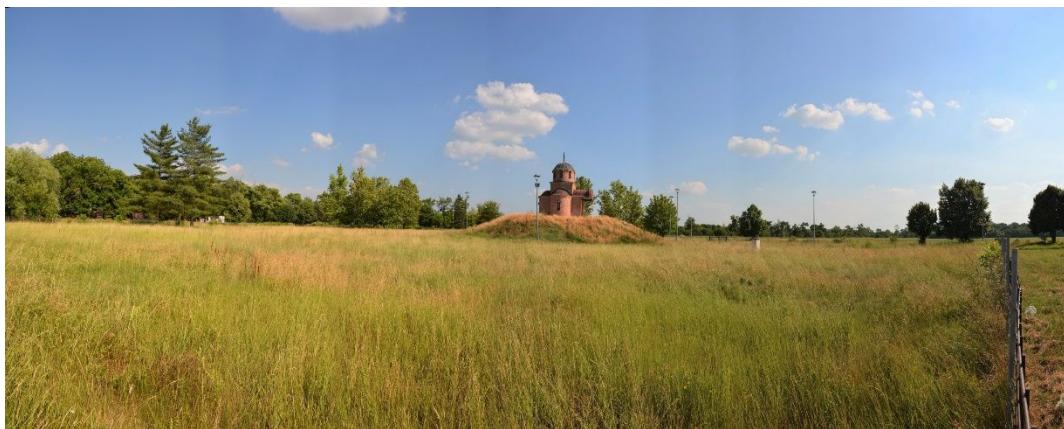


Slika 29 – Konačna dobijena panorama ORB algoritmom za ekstrakciju obeležja

Na datim rezultatima slika moguće je uvideti da svaki od algoritama dobro radi svoj posao, uprkos različitim obeležjima koji se koriste u datim algoritmima. Na svakoj od slika mogu se uočiti mesta spajanja ako se posmatrač dobro zagleda i ako zna da je ovo veštački napravljena panorama. Pored toga, ono što se može uočiti je da ova panorama izgleda u potpunosti verodostojno, bez pojave ikakvih drugi distorzija na slikama, pa bi se tako mogla koristiti i u profesionalne svrhe bez problema. U nastavku će biti prikazani rezultati panorama cilindrične projekcije svakog algoritma, kao i ugrađene funkcije date biblioteke.

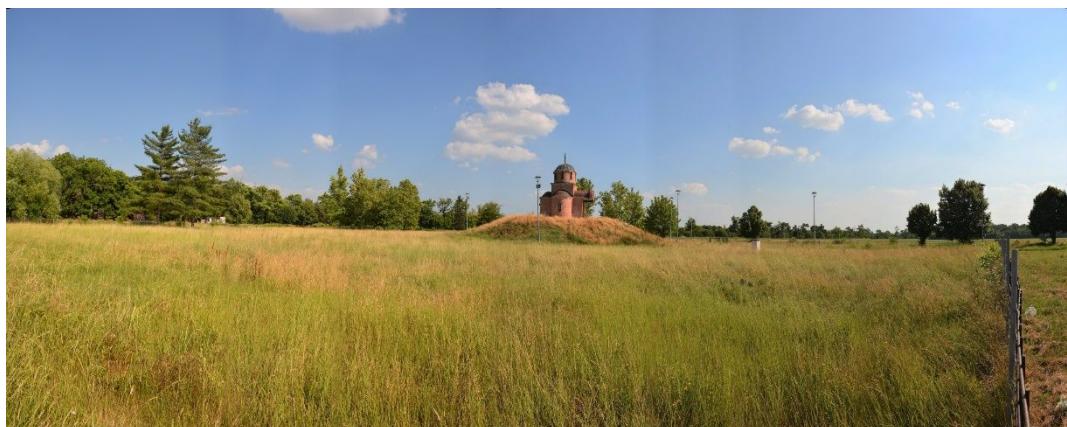
1. primer: Rezultati korišćenih algoritama-cilindrična projekcija

Harisov detektor



Slika 30 – Konačna dobijena panorama Harisovim algoritmom za ekstrakciju obeležja

SIFT algoritam



Slika 31 – Konačna dobijena panorama SIFT algoritmom za ekstrakciju obeležja

SURF algoritam



Slika 32 – Konačna dobijena panorama SURF algoritmom za ekstrakciju obeležja

ORB algoritam



Slika 33 – Konačna dobijena panorama ORB algoritmom za ekstrakciju obeležja

1. primer: Rezultati korišćenih algoritama-ugrađena *OpenCV* funkcija



Slika 34 – Konačna dobijena panorama ugrađenom OpenCV funkcijom

Kao što se može videti na datim rezultatima, i cilindrične projekcije panorame daju dobre rezultate i to kada je u pitanju svaki algoritam. Na Slici 34 može se videti rezultat dobijen ugrađenom funkcijom. Jedina prednost datog rezultata je ta što se na dobijenoj slici ne primećuju toliko porubi na slikama nastali spajanjem slika, odnosno korišćen je veći obim *blending*-a slike. Iz datih rezultata može se zaključiti da svaki od algoritama radi dobro za probleme slika koji imaju akcenat na prirodi, uz malo korigovanje odgovarajućih parametara koji zavise od samih slika. U nastavku će biti prikazani rezultati slika obe projekcije svakog algoritma za preostala dva direktorijuma slika, pa će reči o tome biti nešto više u nastavku.

2. primer: Rezultati korišćenih algoritama-planarna projekcija

Harisov detektor



a)

SIFT algoritam



b)

SURF algoritam



c)

ORB algoritam



d)

Slika 35 – a-d: Rezultati planarne projekcije svih algoritama

Na datim slikama mogu se videti rezultati dobijeni spajanjem slika planarne projekcije. Na prvi pogled može se uočiti da svi algoritmi daju lepe rezultate. Ono što bi se moglo izdvojiti kao jedna od loših stvari je ta što pri spajanju slika planarne projekcije čistom translacijom dolazi do pojave određenih distorzija na slikama kao što je gubitak same geometrije određenog predmeta na spojevima slika što se može uočiti pri vrhu slike kada je u pitanju sama polica. Pored toga algoritmi rade jako dobro, čak iako problemi zasnovani na enterijeru zahtevaju nešto više pažnje, sudeći po tome da se sastoje od dosta detalja i dosta pronađenih obeležja. U nastavku će biti prikazani rezultati dobijeni cilindričnim projekcijama.

2. primer: Rezultati korišćenih algoritama-cilindrična projekcija

Harisov detektor



a)

SIFT algoritam



b)

SURF algoritam



c)

ORB algoritam



d)

Slika 36 – a-d: Rezultati cilindrične projekcije svih algoritama

Na Slici 36 nalaze se rezultati dobijeni za cilindrične projekcije slika. Na datim slikama može se zapaziti da si algoritmi rade dobro i da su date panorame vrlo verodostojne. Ovi rezultati su sami po sebi dosta bolji nego rezultati planarne projekcije zato što je na ovim slikama izvršeno cilindrično uvijanje, pa je sam efekat dobijanja cilindrične projekcije doveo do savršenog poklapanja svih elemenata na slici bez pojave distorzija na vrhu slike kada je u pitanju polica za knjige. Ovi rezultati su skoro identični rezultatu koji daje ugrađena funkcija *OpenCV* biblioteke koja će biti prikazana na sledećoj stranici na Slici 36.

2. primer: Rezultati korišćenih algoritama-ugrađena *OpenCV* funkcija



Slika 37 – Konačna dobijena panorama ugrađenom OpenCV funkcijom

Na datoј slici se vidi da je ugrađena funkcija napravila savršenu panoramu bez pojave ikakvih grešaka koje bi uticale na ocenu panorame. Upoređivanjem prethodnih rezultata sa ovom slikom pronalazi se jako velika sličnost, što govori o tome da implementirani algoritmi rade pravu stvar.

3. primer: Rezultati korišćenih algoritama-ugrađena *OpenCV* funkcija



Slika 38 – Konačna dobijena panorama ugrađenom OpenCV funkcijom

3. primer: Rezultati korišćenih algoritama-planarna projekcija

Harisov detektor



a)

SIFT algoritam



b)

SURF algoritam



c)

ORB algoritam



d)

Slika 39 – a-d: Rezultati planarne projekcije svih algoritama

Na Slici 39 mogu se videti rezultati dobijeni korišćenjem svih algoritama na planarnim projekcijama slika. Uvidom u rezultate može se ustanoviti da svi algoritmi rade skoro podjednako dobro kada je u pitanju rešavanje ovog tipa problema, a to su slike sa urbanim motivima, građevinama i zelenom okolinom. Na ovim slikama planarne projekcija nije izražena pojava distorzija prilikom spajanja slika kao što je to bio slučaj sa primerom prethodnih slika. Kada bi se posmatrač duboko zagledao u sve delove slika, moglo bi se nazreti neke nesavršenosti, ali u globalnom smislu, rezultat ove obrade je jako dobar i ovakva panorama bi se mogla koristiti u nekoj od svojih primena. U nastavku će biti prikazani rezultati cilindrične projekcije.

3. primer: Rezultati korišćenih algoritama-cilindrična projekcija

Harisov detektor



a)

SIFT algoritam



b)

SURF algoritam



c)

ORB algoritam



d)

Slika 40 – a-d: Rezultati cilindrične projekcije svih algoritama

Na Slici 40 prikazani su rezultati slika cilindrične projekcije. Upoređivanjem ovih rezultata sa rezultatom ugrađene funkcije može se zaključiti da su rezultati gotovo identični, što govori o tome da svi implementirani algoritmi rade odlično. Na datim slikama svaki segment izgleda onako kako bi trebalo, što je jednim delom i zasluga cilindrične projekcije slika, pa bi ovakva panorama mogla da se koristi u razne svrhe i primene u kompjuterskoj viziji. Izvodi se zaključak da je implementacija ovih algoritama protekla uspešno i da su njihovim testiranjem dobijeni zadovoljavajući rezultati. Ovi algoritmi mogli bi se nadograditi na razne načine i time još poboljšali performanse, ali o tome će biti malo reči u sledećem poglavlju koje se zove Diskusija, a u nastavku možete videti rezultate ovih algoritama za panorame u 360°.

4. primer: Rezultati korišćenih algoritama-planarna projekcija 360°

Harisov algoritam



Slika 41 – Konačna dobijena panorama Harisovim algoritmom za ekstrakciju obeležja

SIFT algoritam



Slika 42 – Konačna dobijena panorama SIFT algoritmom za ekstrakciju obeležja

SURF algoritam



Slika 43 – Konačna dobijena panorama SURF algoritmom za ekstrakciju obeležja

ORB algoritam



Slika 44 – Konačna dobijena panorama ORB algoritmom za ekstrakciju obeležja

Na datim slikama prikazani su rezultati dobijeni spajanjem slika planarne projekcije različitim obeležjima. Svaki od algoritama radi prilično dobro, sa sitnim distorzijama koje na prvi pogled golim okom nisu uopšte vidljive, što potvrđuje da svaki pristup ispunjava uslove. U nastavku će biti prikazane panorame cilindrične projekcije svakog algoritma.

4. primer: Rezultati korišćenih algoritama-cilindrična projekcija 360°

Harisov algoritam



Slika 45 – Konačna dobijena panorama Harisovim algoritmom za ekstrakciju obeležja

SIFT algoritam



Slika 46 – Konačna dobijena panorama SIFT algoritmom za ekstrakciju obeležja

SURF algoritam



Slika 47 – Konačna dobijena panorama SURF algoritmom za ekstrakciju obeležja

ORB algoritam



Slika 48 – Konačna dobijena panorama ORB algoritmom za ekstrakciju obeležja

Na datim slikama mogu se videti rezultati cilindrične projekcije slika. Svaki od algoritama radi odlično i uvidom u detaljni izgled slike utvrđeno je da su distorzije nastale u planarnim projekcijama ovog puta izostavljene. Rezultati panorama se u velikoj meri poklapaju sa rezultatom dobijenim ugrađenom funkcijom u *OpenCV* biblioteci odakle se može, takođe, zaključiti da ovi algoritmi rade dobro i na panoramama u 360°.

5 DISKUSIJA

Kao što se moglo zaključiti iz prethodno dobijenih rezultata, implementacija ovih algoritama daje zadovoljavajuće rezultate kada su u pitanju obe vrste projekcija. Ono što je već poznato i pomenuto je više puta, dostupan je bio set od 30 slika koje su se koristile u implementaciji algoritma za spajanje slika. Slike su bile podeljene u četiri direktorijuma, u prvom je bilo sedam slika koje većinom predstavljaju elemente prirode i pejzaža, drugom i trećem je bilo po četiri slike koje su predstavljale enterijer i urbane elemente, a u četvrtom je bilo petnaest slika koje su se koristile za izradu panorame u 360 stepeni. Ovakav odabir slika nije bio slučajan. Akcenat je bio na tome da se implementirani algoritam ispita na više tipova slika, odnosno drugačijim tipovima obeležja i situacijama koje se nalaze u samoj slici. Očekivani rezultat ovog algoritma bila je precizno spojena panorama koja od zadatih slika čini jednu jedinstvenu sliku koja kasnije može imati svoju upotrebu. Na osnovu toga dolazi se do zaključka da se implementacija ovog algoritma može okarakterisati kao uspešna, sudeći po rezultatima opisanim u prethodnom poglavlju. Ono što bi mogla biti jedna od mana ovog algoritma je što sama njegova izrada traje otprilike oko 30 sekundi, što je dosta veliki vremenski period ako bi se data vremenska složenost uporedila sa vremenskom složenošću ugrađene funkcije pomoću koje su se, takođe, dobijali rezultati. Sami algoritmi za ekstrakciju obeležja ne traju dugo, vreme izvršavanja je u proseku oko jedne sekunde, što je skroz prihvatljivo, međutim, sami algoritmi poklapanja obeležja i RANSAC algoritma zahtevaju dosta računa i oni su zapravo ti algoritmi koji povećavaju ukupnu vremensku složenost. Takođe, onaj ko bi se bavio ovim algoritmom, potrebno mu je da zna sve žižne daljine prikupljenih slika, što je ponekad malo zametan posao, pogotovu ako ne postoji uslovi za adekvatnu procenu žižne daljine ili nije dostupna specifikacija uređaja iz koje bi se mogli izvaditi odgovarajući zaključci ili ako su preuzete slike sa interneta, bez ikakvog znanja kakvim aparatom su zapravo te slike usnimljene. To nas dovodi do toga da se žižne daljine moraju nagađati, što u većini slučajeva bude jako naporno i dugotrajno. Takođe, algoritam funkcioniše uz definisanje određenih konstanti koje bi se mogle menjati u zavisnosti od trenutnog problema. Pored toga, kako je bitno i kako su i čime date slike prikupljene, bilo bi poželjno da je kvalitet aparata što bolji, kako bi se što tačnije i jasnije mogle prepoznati sve sitnice na slici koje predstavljaju neke bitne faktore. Ono što treba napomenuti je da ovaj algoritam ne radi u realnom vremenu, kao što je već rečeno gore, potrebno mu je neko vreme izvršavanja. Danas je zastupljeno dosta programa i aplikacija koji brzo i efikasno vrše spajanje slika u realnom vremenu. Jedan prost primer takvih algoritama je aplikacija na svakom bolje opremljenom mobilnom telefonu koja je u stanju da snimi panoramu u jednom potezu čak i za celih 360 stepeni svoje ose. Pored toga, ovaj algoritam se koristi i u drugim granama inženjerstva, kao što su arhitektura,

medicina, biologija, geologija i mnoge druge nauke koje u svom glavnom opusu imaju prikupljanje širokih slika za različite potrebe.

Kada je u pitanju poređenje algoritma sa literaturom, [1] *Matthew Brown* i *David G. Lowe* sa univerziteta u Britanskoj Kolumbiji u Vankuveru su se u svom radu bavili sličnim pristupom rešavanju problema. Prednost njihovog algoritma je što je on potpuno automatizovan po pogledu slika. Moguće je ubaciti slike bilo kojim redosledom i algoritam će uvek pronaći odgovarajući par slika koji treba spojiti, pri čemu je omogućeno spajanje slika sa bilo koje strane, leve ili desne. Ova mogućnost nije implementirana u radu, ali je zato omogućeno spajanje slika u pravsu sa leva na desno ili sa desna na levo. Njihov algoritam bazira se na ekstrakciji obeležja *SIFT* algoritmom, dok se u ovom radu ekstrakcija vrši na čak četiri različita načina. Nakon toga se parovi poklapanja nalaze pomoću *KNN* metode. Dodatna stvar kod njihovog algoritma je ta što se nakon *RANSAC* algoritma radi verifikacija parova probabilističkim modelom, što u ovom radu nije slučaj. Takođe, u radu se koristi i *Bundle Adjustment* metoda, dok se u ovom radu pristupalo drugačije. [2] *Natthavut Vong* i *Chatklaw Jareonpona* sa Mahasarakham univerziteta u Tajlandu su u svom radu predstavili rešavanje ovog problema samo za cilindrične projekcije slika, dok se u ovaj rad bazira i na slike planarne i na slike cilindrične projekcije. Pored toga, algoritam za ekstrakciju obeležja koji je korišćen je *SURF* algoritam, dok su u ovom radu korišćena četiri različita algoritma. [3] *Richard Szeliski* i *Heung-Yeung Shum* su u svom radu dali neke teorijske osnove za primenu algoritama korišćenih u ovom radu. Izložen je postupak mapiranja planarnih slika u slike cilindrične i sferične projekcije sa detaljnim postupkom i objašnjenjima, kao i algoritmima za izdvajanje obeležja. Pored toga, dato je više načina kojima se može tačno estimirati žižna daljina koja je od ključne važnosti kada je u pitanju prebacivanje slika u cilindrični i sferični sistem, kao i pojašnjenja kada je u pitanju poravnanje i *blending* slike. Sve su ovo segmenti koji su korišćeni u izradi ovog rada pojedinačno, ali se oni u svom radu nisu bavili konkretnim rešavanjem nekih problema, dok ovaj rad ima praktičnu primenu svih ovih gore pomenutih algoritama.

Pored ovih metoda, u literaturi se mogu naći i mnoge druge implementacije algoritama na bazi spajanja slika sa kojima bi se ovaj rad mogao uporediti, sudeći po tome da se ova oblast neprestano širi i razvija i konstantno dolazi do pronalaženja novih i bržih, boljih algoritama za rešavanje ovakvih i sličnih problema.

6 ZAKLJUČAK

Na osnovu svega izloženog do sada, jasno se može potvrditi da je dati algoritam odgovorio na sve potrebne segmente koji su od ključnog značaja za uspešno rešavanje ovog problema, što i sami rezultati slika pokazuju. Činjenica je da su svi algoritmi za izdvajanje obeležja radili dobro i da se uz pomoć svih tipova dobijenih obeležja mogla uspešno spojiti i dobiti jedna panorama odličnog kvaliteta. U pogledu različitih projekcija, prednost se daje slikama cilindrične projekcije zbog perspektive slika koje su prirodnije išle uz samo uvijanje i spajanje slika, čineći da njihova panorama izgleda baš onako kako treba, kao da je snimljena profesionalno nekim od već implementiranih programa u širokoj upotrebi. Jedno od mogućih unapređenja ovog algoritma je njegova modifikacija i primena u realnom vremenu, čime bi se ovaj algoritam mogao koristiti u razne svrhe, samim tim jer je potražnja za ovakvim implementacijama u realnom vremenu danas jako velika. Takođe, algoritam bi se mogao dalje istražiti po pitanju ostalih 2D i 3D transformacija kao što su rotacija, skaliranje, smicanje, afina transformacija i projektivna transformacija i mnoge druge. Takođe, mogli bi se koristiti još neki od algoritama izdvajanja obeležja, pa da se ispituju performanse primene takvih algoritama u ove svrhe. Pored posmatranih projekcija slika, postoje i druge projekcije kao što su sferna projekcija ili poznata *fish eye* projekcija koje, takođe, imaju široku primenu, a mogla bi se nadovezati na ovaj rad. Takođe, zanimljiv pristup rešavanju ovog problema jeste spajanje slika u različitim delovima dana, odnosno danju i noću, kako su neki od implementiranih algoritama invarijantni na osvetljenost, ovaj postupak je omogućen. Ovakav vid algoritma može se primeniti i na video snimke, ne samo na slike, što bi bio deo implementacije algoritma u realnom vremenu pomenute ranije. Na samom kraju, ovakva koncepcija algoritma mogla bi se iskoristiti za treniranje neuralne mreže koja bi na osnovu datih obeležja i ostalih osobina mogla efikasno da vrši spajanje slika, međutim, ovo predstavlja mnogo kompleksniji problem koji zahteva dosta veći set podataka za obučavanje.

Svedoci smo sve većeg razvijanja tehnologije poslednjih godina, a pogotovu kada su u pitanju razni programi za spajanje slika, počevši od mobilnih telefona kod kojih se danas sve više koristi ovakav tip fotografisanja, do snimanja geografskih lokacija iz različitih perspektiva i koordinatnih sistema, izrade online mapa za prikaz i navigaciju, prikaza slika u krugu od 360 stepeni i mnogim drugim sferama kompjuterske vizije koja tek doživljava svoj procvat.

7 LITERATURA

- [1] Matthew Brown, David G. Lowe, "Automatic Panoramic Image Stitching using Invariant Features", International Journal of Computer Vision 74(1), 59–73, 2007. godina
- [2] Natthavut Vong, Chatklaw Jareonpon, "Multi image stitching with cylindrical surface base on local feature matching for solving the distortion problem", International Conference on Intelligent Systems and Image Processing, 2015. godina
- [3] Richard Szeliski, Heung-Yeung Shum, "Creating Full View Panoramic Image Mosaics and Environment Maps", Association for Computing Machinery, 2020. godina
- [4] dr Veljko Papić, Auditorna predavanja sa predmeta Kompjuterska vizija "Slaganje slika", šifra predmeta 13M051KV, 2021. godina
- [5] David G. Lowe. "Distinctive image features from scale-invariant keypoints." IJCV 60 (2), pp. 91-110, 2004. godina
- [6] Zongyuan Zhu, Guicang Zhang, Hongjie Li, "SURF feature extraction algorithm based on visual saliency improvement", International Journal of Engineering and Applied Sciences (IJEAS) ISSN: 2394-3661, Volume-5, Issue-3, 2018. godina
- [7] dr Veljko Papić, Auditorna predavanja sa predmeta Kompjuterska vizija "Lokalna obeležja", šifra predmeta 13M051KV, 2021. godina

PRILOG A

Propratni delovi master rada koji su korišćeni u izradi dati su u prilogu sledećim redosledom:

1. programski kod u jeziku Python 3.7,
2. slike korišćene u izradi master rada,
3. prezentacija master rada,
4. prikupljeni set slika,
5. CD sa prethodno navedenim prilozima,
6. i drugo.