Detekcija grana za rezidbu na RGB-D slikama

Bošnjak, Andrej

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I
INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Diplomski sveučilišni studij Računarstvo
Izborni blok Robotika i umjetna inteligencija

Detekcija grana za rezidbu na RGB-D slikama

Diplomski rad

Andrej Bošnjak

Osijek, 2024.

Sadržaj

1. UVOD

Uvođenje robotizacije u različite grane poljoprivrede, kao što su voćarstvo, vinogradarstvo, povrtlarstvo i slično, postaje sve važnija tema s obzirom na rastuće potrebe za povećanjem efikasnosti i smanjenjem troškova proizvodnje zbog sve većeg broja stanovništva, većih potreba za hranom, manjeg uzgoja hrane po kućanstvima i sve većeg oslanjanja na kupovinu hrane. Jedan od ključnih procesa u poljoprivrednoj proizvodnji i održavanju voćnjaka je rezidba grana, koja tradicionalno zahtijeva značajnu i obučenu radnu snagu. Rezači moraju poznavati pravila rezidbe i specifične tehnike, što rezidbu u velikim voćnjacima čini skupom i dugotrajnom aktivnošću. Ovaj proces je neophodan za održavanje zdravlja biljaka i poboljšanje prinosa voćnjaka, što opravdava potrebu za njegovom automatizacijom. Robotizacija rezidbe uključuje preciznu detekciju grana za rezidbu na slici pomoću računalnih algoritama, što je predmet aktualnih istraživanja. Pri tome, jedan od ključnih izazova je sunčeva svjetlost, odnosno promjenjivo osvjetljenje u voćnjacima, koje može drastično utjecati na kvalitetu slike. Kvaliteta senzora i kamera također igra važnu ulogu, jer slabiji senzori mogu rezultirati mutnim ili nejasnim slikama i gubitkom informacija na slici. Osim toga, razvoj pouzdanih algoritama zahtijeva opsežne i kvalitetne podatkovne skupove za treniranje modela strojnog učenja.

Današnji tehnološki razvoj omogućuje značajan napredak u robotizaciji poljoprivrednih poslova. Na primjer, napredak u umjetnoj inteligenciji i strojnome učenju omogućio je razvoj sofisticiranih algoritama za prepoznavanje objekata, što je ključno za zadatke poput precizne rezidbe grana. Razvoj autonomnih vozila i robota specijaliziranih za poljoprivredne zadatke smanjio je potrebu za ljudskom intervencijom u mnogim procesima. RGB-D slike, koje kombiniraju RGB s dubinskim podacima pružaju bogat skup informacija potrebnih za preciznu detekciju grana za rezidbu. RGB-D slike, naspram RGB slika, pogodnije su za korištenje u detekciji grana za rezidbu i robotizaciji rezidbe zbog dodatnih informacija koje pruža dubina na slici, npr. za precizno određivanje pozicije točke rezidbe grane i precizno računanje putanje robota do te točke. Korištenjem informacija o dubini, moguće je razviti napredne algoritme za detekciju grana i pronalazak mjesta reza grane, omogućavajući robotima da se samostalno pozicioniraju i izvrše rezidbu grana.

Prepoznavanje objekata i njihovih dijelova uobičajeno se na slici provodi pomoću nekih tradicionalnih metoda strojnog učenja za detekciju i prepoznavanje objekata, ali u posljednje vrijeme sve češće pomoću neuronskih mreža. Za složene probleme poput detekcije grana, neuronske mreže predstavljaju logičan izbor. Općenito, za učenje neuronske mreže potrebno je puno označenih podataka. Kako bi se neuronska mreža naučila detektirati grane za rezidbu, mora se istrenirati na slikama drveća s označenim granama za rezidbu. S obzirom da je takvih podatkovnih skupova malo, u ovom radu predstavljen je prikupljeni podatkovni skup RGB i dubinskih slika voćaka prije i poslije rezidbe. Takav podatkovni skup se može koristiti za razvijanje algoritama i treniranje neuronskih mreža za detekciju grana.

Ideja ovog diplomskog rada je kreirati podatkovni skup RGB-D slika drveća prije i nakon rezidbe te na temelju njih implementirati algoritam za automatsko označavanje grana za rezidbu. U ovom se radu na temelju snimljenih RGB-D slika kreiraju oblaci točaka pomoću kojih se rekonstruira 3D model drveta. Razlika preklopljenih 3D modela drveća prije i nakon rezidbe predstavlja odrezane grane, to jest točke koje pripadaju odrezanim granama. Te točke predstavljaju referentne vrijednosti na temelju kojih se, uz 3D modele drveća prije rezidbe, može trenirati neuronska mreža. Zatim se kreira neuronska mreža temeljena na već postojećim arhitekturama za rad sa oblacima točaka. Ta neuronska mreža za zadatak ima klasificirati svaku točku 3D modela drveća, pružajući kao izlaz oznake pripada li točka odrezanoj grani ili ne. Ovakva mreža omogućava automatizaciju procesa označavanja grana za rezidbu.

Rad je strukturiran na sljedeći način. U drugom poglavlju dan je pregled postojećih

metoda i algoritama za detekciju grana, opisana je njihova primjena te su navedeni nedostatci metoda. U trećem poglavlju su detaljno opisani način prikupljanja podatkovnog skupa slika, korištena oprema i tehnologije za prikupljanje slika te struktura prikupljenog podatkovnog skupa. U četvrtom poglavlju detaljno je opisan svaki korak razvijene metode za rekonstrukciju 3D modela drveća na temelju RGB-D slika i označavanje orezanih grana. Također u tom poglavlju detaljno je opisana arhitektura na kojoj se temelji kreirana neuronska mreža. U petom poglavlju prikazana je osnovna provjera valjanosti metode, njena evaluacija te mogući problemi prilikom rekonstrukcije stabla. Nadalje, prikazana je i evaluacija istrenirane mreže na prikupljenom podatkovnom skupu. Zaključak je dan u šestom poglavlju.

2. PREGLED PODRUčJA DETEKCIJE GRANA NA RGB-D SLIKAMA

U ovom poglavlju pružen je uvid u postojeća rješenja detekcije grana na RGB-D slikama. Istražena su rješenja koja koriste dubinske informacije zbog njihove važnosti u razvoju algoritama za robotsku manipulaciju, tojest robotiziranu rezidbu. Također je opisana njihova primjena i navedeni su njihovi nedostatci.

Akbar i suradnici u radu *A Novel Benchmark RGBD Dataset for Dormant Apple Trees and Its Application to Automatic Pruning* [1] predstavljaju podatkovni skup od 9 voćaka jabuke. Podatkovni skup je prikupljen korištenjem Kinect2 senzora. Osim RGB i dubinskih slika voćaka, podatkovni skup sadržava i označene slike koje služe kao temeljna istina, izmjerene promjere glavnih grana i relativnu udaljenost između para primarnih grana. Ovaj podatkovni skup može predstavljati osnovne referentne vrijednosti za procjenu različitih 3D algoritama rekonstrukcije i modeliranja stabala voćaka.

Zhang i suradnici u radu *Branch detection for apple trees trained in fruiting wall architecture using depth features and Regions-Convolutional Neural Network (R-CNN)* [2] razvili su metodu za detekciju grana na stablima jabuke koja se temelji na regionalnoj konvolucijskoj neuronskoj mreži (R-CNN). Koristili su Microsoft Kinect v2 kameru za prikupljanje RGB slika, slika u pseudo boji (engl. *pseudo-color images, PCI*) i dubinske slike u prirodnom okruženju. R-CNN mreža se sastoji od poboljšane AlexNet mreže i učena je za detektiranje grana stabla jabuke. U ovom se radu uspoređuju rezultati učenja na PCI i PCI-D slikama. Kao što je i očekivano, rezultati su bili bolji na slikama koje sadrže i informacije o dubini. Dobiveni rezultati od 92% za prosječni odziv i 86% za prosječnu točnost te prosječni korelacijski koeficijent r između detektiranih i referentnih grana od 0.91 pokazuju obećavajući napredak u detekciji i lokaliziranju grana stabla u voćnjaku, što služi kao prvi korak u automatizaciji berbe, a također može služiti i za automatizaciju orezivanja. Najveći utjecaj na preciznost estimacije su imale male grančice koje su negativno utjecale na estimaciju položaja središta grane.

Također kao prvi korak u automatizaciji postupka orezivanja stabla, rad *A Method for Three-Dimensional Reconstruction of Apple Trees for Automated Pruning* [3] predstavlja metodu za 3D rekonstrukciju stabla jabuke koja se temelji na tehnici oblik-iz-siluete, predstavljena u *Shape-From-Silhouette Across Time Part I: Theory and Algorithms* [4]. 3D rekonstrukcija se postiže iz slika slikanih iz više orijentacija i kutova, koji su prethodno definirani. Kamera je postavljena na pan-tilt mehanizmu, gdje se točno određuju položaji kamere prilikom snimanja slika. Ovim putem se dobije točna transformacijska matrica koja olakšava spajanje više oblaka točaka u jedan cjeloviti, koji predstavlja stablo. Kako bi algoritam bio računalno učinkovit, korišten je algoritam stanjivanja za skeletonizaciju 3D oblaka točaka. Pomoću stanjenih oblaka točaka, predloženom metodom je postignuta 100% točnost za detektiranje debla i 77% točnost za detektiranje grana. Metoda je također detektirala 23% lažno pozitivnih i lažno negativnih rezultata. Ovakva metoda se pokazala učinkovitom u rekonstrukciji stabla, no zahtjeva predodređene položaje kamere, koje nisu uvijek moguće u stvarnim okruženjima. Ovom metodom, isti autori u radu *Identification of pruning branches in tall spindle apple trees for automated pruning* [5] stvaraju rekonstruirane 3D modele stabla jabuke. S tako stvorenim modelima stvaraju i metodu za određivanje grana za rezidbu, uz pojednostavljen proces rezidbe grana. Taj pojednostavljeni proces podrazumijeva održavanje određenog razmaka grana i održavanje specificirane duljine grana. Performanse algoritma su optimizirane korištenjem skupom podataka za učenje od 10 stabala kako bi se postigla razina rezidbe ljudskog radnika. Uz odabrani maksimalni razmak grana od 28 cm i odabrane minimalne duljine grana od 20 cm, algoritam predlaže rezidbu oko 20% grana, dok radnici predlažu udio orezanih grana od 22%. Algoritam i ljudsko orezivanje grana rezultiralo je sličnim razmakom između grana. Korijen

srednje kvadratne devijacije između ljudske identifikacije grana za rezidbu i one dobivene algoritmom je 13%, odnosno algoritam uspješno identificira grane za rezidbu. Jedno od poboljšanja koje spominju autori je da se koristi više perspektiva stabla s različitih položaja kamere za precizniju rekonstrukciju stabla te poboljšanje algoritma da izbacuje umrtvljene grane i grane sa velikim promjerom. S obzirom na sustav na koji je postavljena kamera, ne postoji sloboda micanja kamere, što dovodi do manjka robusnosti sustava na različite oblike i veličine stabala.

Wang i Zhang u svom radu *Three-Dimensional Reconstruction of a Dormant Tree Using RGB-D Cameras* [6] istražuju mogućnost rekonstrukcije stabla iz samo dvije RGB-D slike. Korištene su dvije RGB-D kamere postavljene u predodređene položaje. Na ovaj način se lako izračuna potrebna transformacijska matrica korištena u spajanju oblaka točaka. Njihov rad je pokazao zadovoljavajuće rezultate u kontroliranim uvjetima u laboratoriju, no također je i pokazano najveće ograničenje RGB-D kamera: njihova osjetljivost na sunčanu svjetlost. Rekonstrukcija stabla snimljenih u voćnjaku tijekom sunčanih dana se pokazala gotovo nemogućom. Suprotno tomu, slike snimljene tijekom noći pokazale su dovoljno dubinske informacije za rekonstrukciju, ali je informacija o boji stabla bila izgubljena. Kako bi se riješili postojeći problemi, autori predlažu unaprjeđenje RGB-D kamera, ali i mogućnost smanjivanja utjecaja sunčeve svjetlosti na sustav kao integrirani dio automatiziranog stroja za rezidbu. Budući da boja grana nije potrebna informacija za detektiranje grana za rezidbu te ako boja nije potrebna za drugi dio sustava, autori predlažu rad sustava u noćnim satima.

Yuxing i suradnici u radu *Three-dimensional model construction method and experiment of jujube tree point cloud using Alpha-shape algorithm* [7] predlažu unaprijeđen algoritam za registraciju oblaka točaka temeljen na iterativnom algoritmu najbliže točke (engl. *Iterative Closest Point* ICP). U radu je također korištena RGB-D kamera za snimanje slika u boji i dubinskih slika, iz kojih je generiran oblak točaka. U oblaku točaka su uklonjene stršeće vrijednosti te je uklonjena pozadinska smetnja. Umjetne oznake u obliku crvenih lopti blizu korijena stabla korištene su za određivanje inicijalne registracije. Iz inicijalne registracije se izračuna vektor normale i zakrivljenost oblaka točaka kako bi se izračunali parovi točaka. U konačnici je korišten ICP algoritam za precizno preklapanje oblaka točaka i rekonstrukciju stabla. Prosječna rezultirajuća pogreška registracije je 0.76 cm, a relativna pogreška rekonstruirane vrijednosti je kontrolirana unutar 7%.

Yaqoob Majeed i suradnici u radu *Apple Tree Trunk and Branch Segmentation for Automatic Trellis Training Using Convolutional Neural Network Based Semantic Segmentation* [8] za cilj imaju segmentirati stablo na deblo i grane pomoću konvolucijske neuronske mreže. Za prikupljanje podataka koriste Kinect V2 senzor koji ima mogućnost snimanja RGB i dubinske slike, kao i oblak točaka. Pozadinsku smetnju filtriraju pomoću praga za dubinu. Koristili su konvolucijsku neuronsku mrežu SegNet koju su razvili autori Vijay Badrinarayanan i suradnici, predstavljenu u radu *SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation* [9]. Za treniranje su koristili 210 ručno označenih slika. Na svakoj slici su označene klase: pozadina, deblo i grane. Skup podataka za učenje se sastojao od 90 slika. Postigli su točnost segmentacije za deblo od 92% i točnost segmentacije za grane od 93%. Za evaluaciju su također koristili metriku presjek-kroz-unija (engl. *intersection-over-union* IoU) te postigli rezultat od 59% za deblo i 44% za grane. Boundary-F1 score metrika izravno povezuje točnost granica segmentirane regije je 93% za deblo, odnosno 88% za grane. Ovakva metoda semantičke segmentacije stabla na deblo i grane predstavlja temelj za razvoj automatiziranog sustava za rezidbu grana.

Noha M. Elfiky i drugi u radu *Automation of dormant pruning in specialty crop production: An adaptive framework for automatic reconstruction and modeling of apple trees* [10] koriste Kinect2 senzor za prikupljanje dvije slike s prednje i stražnje strane stabla, s obzirom na to da u modernim voćnjacima nije moguće skenirati cijelo stablo u krugu od 360°, zbog potpornih žica između betonskih stupova na kojima rastu voćke. Iz te dvije slike, pomoću ge-

ometrijskih značajki temeljenih na kosturu modela te u konačnici dorađivanje i usklađivanje sa *ICP* algoritmom, rekonstruiraju oblak točaka cijelog stabla. Na dobivenom 3D modelu provode predloženu *CLAM* (engl. *Circle-based-Layer-Aware Modeling*) tehniku za segmentiranje glavnih grana, označenih u radu s "PB" (engl. *primary branch*), od debla stabla. Glavna značajka po kojoj se određuje točka rezanja grane jest promjer glavnih grana. Za evaluaciju su koristili oznake radnika u voćnjacima za točke rezanja te fizičko mjerenje promjera grane u toj točki i uspoređivanje s debljinom grane u točki rezanja koju je algoritam odredio. Točnost određivanja točke rezidbe se očituje kroz metriku *PPRA* (engl. *Pruning Point Radius Accuracy*) uz toleranciju od 2 mm. *PPRA* za sintetičke modele je 100%, dok je za modele prikupljene u vanjskom voćnjaku 78%. Također točnost određivanja glavnih grana je 92.2%. Za proširenje ovakvog sustava predlažu automatiziranje uklanjanje susjednih stabla i grana iz slike od stabla od interesa. Također predlažu adaptivno određivanje pragova za razne dijelove algoritma korištenih u njihovom sustavu.

3. BRANCH - PODATKOVNI SKUP RGB-D SLIKA VOćAKA PRIJE I NAKON REZIDBE

U ovom je poglavlju detaljno opisan podatkovni skup korišten u ovom radu i način njegova prikupljanja. Prikupljeni podatkovni skup sadrži RGB i dubinske slike stabala voćaka iz kojih se dobivaju oblaci točaka primjenom razvijenog algoritma. Zbog nedostatka postojećih podatkovnih skupova specifičnih za ovaj zadatak, bilo je nužno prikupiti vlastiti skup podataka kako bi se omogućio razvoj i evaluacija algoritama strojnog učenja za ovu specifičnu primjenu. Ovaj podatkovni skup može služiti za daljnje razvijanje sličnih algoritama i njegovo prikupljanje predstavlja značajan doprinos jer takvi podatkovni skupovi nisu široko dostupni ili prethodno dokumentirani.

3.1. Opis voćnjaka

Podatkovni skup je prikupljen u suradnji s Poljoprivrednim Institutom u Osijeku koji je omogućio pristup voćnjaku, koji se nalazi u predgrađu Osijeka, prije i nakon rezidbe. Voćnjak, prikazan na slikama 3.1 i 3.2, je rešetkastog oblika (engl. *trellis orchard*), tj. postavljeni su betonski stupovi između kojih su provučene dvije metalne žice na visinama od otprilike 50 cm i 150 cm od tla. Na te je žice povezano svako stablo. Između dva betonska stupa posađeno je 10 stabala voćki s međusobnim razmakom od cca 100 cm. Podatkovni skup čine snimljena stabla kruške, sorte viljamovka i sadrži 184 slika stabala prije i poslije rezidbe u sumrak i 32 stabla u zoru.

Slika 3.1: *Voćnjak prije rezidbe.*

Slika 3.2: *Voćnjak nakon rezidbe.*

Stabla su slikana prije i poslije rezidbe kako bi, nakon 3D rekonstrukcije stabla, bilo moguće napraviti razliku u oblacima točaka i na taj način dobiju vidljive točke koje su orezane. Slike neposredno prije rezidbe su slikane u kasnu zimu (5.3.2024.-8.3.2024.), dok su slike neposredno nakon rezidbe slikane početkom proljeća (19.3.2024.-22.3.2024.). Cilj je bio snimiti stabla u što kraćem vremenskom razmaku kako bi se snimka istog stabla prije i poslije

rezidbe razlikovala samo u odrezanim granama, a da se izbjegnu druge promjene poput listanja i cvjetanja. Rezidba je provedena pred samo cvjetanje pa se na slikama nakon rezidbe ipak vide procvjetale grane. Algoritam bi stoga trebao biti robustan na ovu pojavu. Točni datumi rezidbe obično variraju iz godine u godinu, ovisno o vremenskim uvjetima i fazama razvoja stabala. Slike su slikane u zoru i sumrak kako bi se izbjegla direktna sunčeva svjetlost koja ometa dubinski senzor. Način na koji sunčeva svjetlost ometa senzor je detaljnije opisano u sljedećem potpoglavlju.

3.2. Senzor

Za prikupljanje podatkovnog skupa korištena je ASUS Xtion PRO Live kamera, prikazana na slici 3.3, koja ima mogućnost slikanja RGB i dubinskih slika. Njene specifikacije dane su u tablici 3.1

Udaljenost korištenja	0.8 m do 3.5 m
Vidno polje	58°H, 45°V, 70°D (vodoravno, okomito, dijagonalno)
Veličina dubinske slike	640*480
Rezolucija	1280*1024
Dimenzije	18 x 3.5 x 5 inča
Softver	Kompleti za razvoj softvera (OPEN NI SDK u paketu)

Tablica 3.1: *Specifikacije ASUS Xtion PRO Live kamere*

Za potrebe snimanja voćki, najveći nedostatak ove kamere, kao i većine dubinskih senzora, je to što je primarno namijenjena za upotrebu u unutarnjim prostorima te je jako osjetljiva na izravnu sunčevu svjetlost. Zbog toga nije moguće dobiti dubinsku sliku koja sadrži informacije o dubini ako sunce direktno obasjava objekt. Zato je prikupljanje podatkovnog skupa provedeno u zoru ili sumrak, kada ima dovoljno svjetlosti da bi RGB slike bile vidljive, a dovoljno malo da se može dobiti i kvalitetna dubinska slika.

Slika 3.3: *ASUS Xtion PRO Live kamera.*

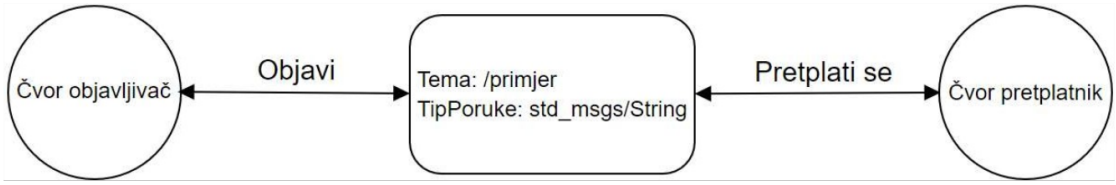
3.3. Programski dio

S obzirom na to da ASUS kamera ima ugrađeni komplet za razvoj softvera OpenNI, korišten je Robotski operacijski sustav (ROS) za pokretanje kamere. Implementacija u ROS-u služi za jednostavnu integraciju ovog programskog dijela u cjeloviti robotski sustav koji će se sastojati od robotske ruke, kamere, alata za rezidbu i mobilne platforme. Takvi cjeloviti robotski sustavi su najčešće međusobno povezani preko Robotskog operacijskog sustava. ROS je standardni okvir za programiranje robota jer se unutar njega lako integriraju programski kodovi i hardver, roboti, senzori i aktuatori [11]. Programski kod koji dohvaća informaciju iz kamere, strukturira podatkovni skup i sprema slike napisan je u obliku Python skripte. Prilikom inicijalizacije programa korisnika se pita za unos svih potrebnih informacija za strukturiranje podatkovnog skupa i tako se stvaraju datoteke u koje će se spremati slike. U primjeru koda 1 vidljivo je spajanje svih tih varijabli u cjelovitu putanju za spremanje podatkovnog skupa.

Primjer koda 1: *Spajanje svih varijabli u potpunu putanju*

```
1 self.rgb ath = os.path.join(self.save root_path, self.camera , self.pruned, self.time of day,
      ↪ self.obj, self.angle, 'color')
2 self.depth ath = os.path.join(self.save_root_path, self.camera_, self.pruned, self.time of
      ↪ day, self.obj, self.angle, 'depth')
```

Kako se kamera pokreće s OpenNI paketom, ona svoje informacije objavljuje u obliku ROS tema (engl. *topic*) kako bi ostali čvorovi (engl. *nodes*) mogli komunicirati i pretplatiti se na tu temu. Zbog toga je potrebno stvoriti ROS paket unutar kojeg se pokreće Python skripta i stvara novi čvor koji je pretplaćen na temu kamere. Pojednostavljeni prikaz čvorova i pretplatnika vidljiv je na slici 3.4.



Slika 3.4: *Pojednostavljeni prikaz čvorova i tema.*

Naredbe s kojima se pretplaćuje na čvor od kamere, koje se nalaze unutar Python skripte za snimanje slika, vidljive su u primjeru koda 2.

Primjer koda 2: *Pretplaćivanje na čvor od kamere*

```
1 self.image_sub_depth = message filters.Subscriber('/camera/depth/image-raw', Image)
2 self.image_sub_rgb = message_filters.Subscriber('/camera/rgb/image_raw', Image)
3
4 ts = message_filters.ApproximateTimeSynchronizer([self.image_sub_depth, self.image sub
      ↪ rgb], 1, 0.1, allow headerless=True)
5 ts.registerCallback(self.callback)
```

Za dohvaćanje slika s čvora kamere koristi se biblioteka OpenCV2. S obzirom na to da s čvora kamere slike dolaze u obliku *imgmsg*, potrebno ih je pretvoriti u cv2 objekt. Za to se koristi CvBridge objekt koji je inicijaliziran na početku skripte. U primjeru koda 3 vidljiv je dio koda koji pretvara informacije s čvora u prikladan oblik.

Primjer koda 3: *Pretvaranje informacije u prikladan oblik*

```
depth_image = self.bridge.imgmsg to cv2(depth data, '16UC1')
rgb_image = self.bridge.imgmsg to_cv2(rgb data, 'bgr8')
```

Varijable depth data i rgb data se kontinuirano dohvaćaju s čvora kamere. U konačnici, na pritisku tipke razmaka, spremaju se slike pomoću *cv2.imwrite()* funkcije. U kodu postoje i naredbe za kontrolu toka programa za stvaranje nove datoteke za novo drvo (tipka q), u slučaju mijenjanja kuta (tipka k) i ponovna inicijalizacija programa za slikanje novog reda stabala (tipka n). Cijeli tok programa vidljiv je u primjeru koda 4.

Primjer koda 4: *Kontrola toka programa*

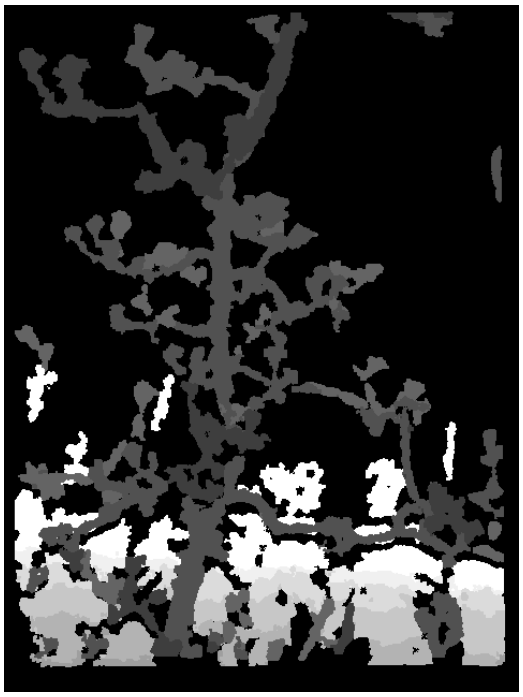
```
if key == 32: # spacebar entered
    cv2.imwrite(self.rgb ath + '/(}.png'.format(self.counter), rgb image)
    cv2.imwrite(self.depth ath + '/(}.prig'.format(self.counter), depth image)
    self.counter += 1
    print('Saved!')
if key == ord('n'):
    ## APT yes–no promt yes = ('yes','y', 'ye', ''} no = ('no','n'}
    choice = input("Select new row?-(Will reset current tree counter to 0!)-[Yes/no]").
        ↪ lower()
    if choice in yes:
        self.row number = input("Select current tree row:") self.tree number = 0
        self.initialize('tree_%s_%s_%04d' % (self.row_number, self.tree_sort, self.tree
            ↪ number)) #format tree red sorta 00xx
    elif choice in no: print("Cancelled")
else:
    print("Cancelled")

if key == ord('k'):
    self.initialize(self.obj, self.angle_no + 1)
if key == ord('q'):
    self.tree number += 1
    self.initialize('tree_%s_%s_%04d' % (self.row_number, self.tree_sort, self.tree number)
        ↪ ) #format tree red sorta 00xx
```

3.4. Opis podatkovnog skupa

Svako stablo je snimljeno iz nekoliko različitih kutova snimanja kako bi na slici bilo vidljivo što više grana te kako bi se integracijom slika dobio cjeloviti 3D model stabla. Za većinu stabala postoje minimalno tri slike na različitim visinama ispred stabla te po još dvije s bočnih strana stabla. Što je stablo veće, to je snimljeno više slika kako bi se obuhvatilo cijelo stablo. S obzirom na to da su stabla gusto posađena, kamera je prilikom snimanja stavljena u vertikalni položaj kako bi na slici bilo vidljivo samo jedno stablo u tom redu. Primjeri jedne (rotirane) RGB slike i njene korespodentne dubinske slike vidljive su na slikama 3.5 i 3.6.

Podatkovni skup sadrži ukupno 184 stabla slikanih prije i poslije rezidbe u sumrak te 32 stabla slikanih prije i poslije rezidbe u zoru. Neka su stabla slikana i ujutro i navečer. Manji je broj slikanih stabala u zoru zato što je vremenski interval u kojemu su RGB slike vidljive, pri čemu na dubinskim slikama postoje informacije o granama, puno manji u zoru nego u sumrak.

Slika 3.5: *Primjer RGB slike stabla.*



Slika 3.6: *Primjer dubinske slike stabla.*

U prosjeku svako stablo ima između 10 i 15 slika, a minimalno 6. Kao polazište za razvoj i evaluaciju algoritma za 3D rekonstrukciju stabla, korišteno je 70 slika stabla prije i poslije rezidbe slikanih u sumrak. Podatkovni skup je strukturiran pomoću grananja direktorija koji opisuju redom:

- Kojom kamerom je snimano (asus)
- Prije ili poslije rezidbe (B / A) (engl. *Before / After*)
- Doba dana (E / M) (engl. *Evening / Morning*)
- Red stabla, sorta i indeks stabla oblika: tree_row_sort_index (npr. tree_1_V_0001)
- Kut snimanja (angle0)
- RGB ili dubinska slika (engl. *color / depth*)

Primjer putanje RGB slike za stablo 35 slikano u zoru prije rezidbe je:
`asus/B/M/tree_1_V_0035/angle0/color/0.png`

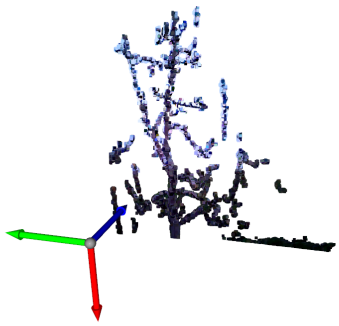
Unutar direktorija nalaze se slike u .png formatu koje su nazvane rednim brojem kojim su slikane (npr. 0.png, 1.png, 2.png). Svaka RGB slika u direktoriju s RGB slikama sadrži svoju korespodentnu dubinsku sliku istog naziva u direktoriju s dubinskim slikama.

4. DETEKCIJA GRANA ZA REZIDBU

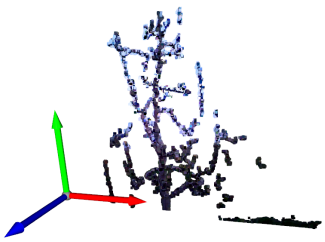
U ovom poglavlju je detaljno opisan postupak rekonstrukcije stabla te označavanje grana za rezidbu. Koraci rekonstrukcije podrazumijevaju stvaranje oblaka točaka iz RGB i dubinskih slika, globalno preklapanje oblaka točaka te poravnavanje ICP algoritmom. Ti koraci se ponavljaju za sve odabrane slike. 3D model stabla je potreban zbog preciznosti određivanja točke rezanja. Robotskoj ruci koja bi automatski orezivala grane je potreban točan položaj te točke u prostoru kako bi se alat za rezanje grana mogao adekvatno pozicionirati.

4.1. Rekonstrukcija modela stabla

Iako je uslikano u prosjeku 10 RGB-D slika svakog stabla, za potrebe rekonstrukcije su korištene samo četiri RGB i dubinske slike, koje sadrže dovoljno informacija. Odabir slika se vrši manualnom inspekcijom svake slike i odabiru se one s najviše vidljivih grana tog stabla i najmanje vidljivih grana drugih stabala. Primjer jednog stabla i odabranih slika za rekonstrukciju vidljiv je na slici 4.9. Rekonstrukcija stabla je izrađena pomoću programskog jezika Python i Open3D Geometry biblioteke [12]. Prvi korak u rekonstrukciji je stvaranje oblaka točaka na temelju svake slike. Oblak točaka se stvara na temelju RGB-D slike i intrinzičnih parametara kamere, pomoću ugrađenih funkcija *RGBDImage.create_from_color_and_depth* i *PointCloud.create_from_rgbd_image* iz Open3D biblioteke. Zatim se cijeli oblak točaka rotira oko Z i Y osi kako bi se poravnala negativna Y os s gravitacijom, kao što je vidljivo na slikama 4.7 i 4.8.



Slika 4.7: *Stvoreni oblak točaka u originalnoj rotaciji.*



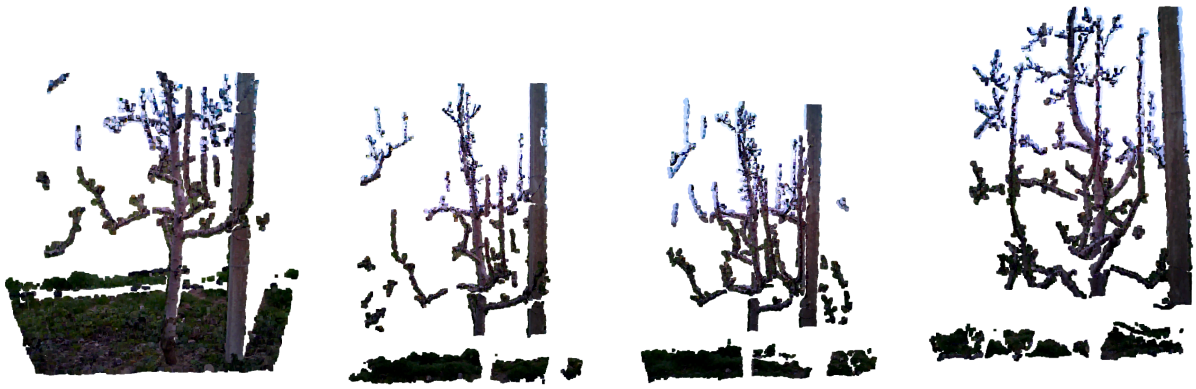
Slika 4.8: *Oblak točaka nakon rotacije oko Z i Y osi.*

Ovaj korak je potreban jer su slike uslikane vertikalno i s tim koracima stablo se rotira u tzv. prirodni položaj, tj. uspravno prema gore. Na taj način oblak točaka postaje vizualno intuitivan i pregledan što olakšava vizualnu inspekciju u daljnjim koracima. Primjer dobivenih obojanih oblaka točaka korištenjem ugrađenih funkcija vidljiv je na slici 4.10. Sljedeći korak podrazumijeva izoliranje promatranog stabla tako da se ukloni sve ono što ne pripada tom stablu. Najčešće u oblaku točaka su vidljive grane drugih stabla i trava. Svaki oblak točaka se ručno analizira i odredi postoji li trava ili ne. Ukoliko postoji trava, uklanja se sa slike primjenom K-means algoritma s dva klastera. Primjer odvajanja trave od ostatka stabla vidljiv je na slici 4.11. Prilikom odabira slika, odabiru se one koje ne sadrže točke grana drugih stabla, ili ih ima dovoljno malo da su zanemarive. Točke betonskih stupova nije moguće ukloniti jer su stabla fizički preblizu tih stupova, tj. u većini slučajeva se i međusobno dodiruju. Ali s obzirom da su betonski stupovi stacionarni i ne mijenja im se oblik, ne predstavljaju problem

prilikom rekonstrukcije niti prilikom označavanja grana za rezidbu. Zbog toga se betonski stupovi ignoriraju i rekonstrukcija svakog desetog stabla sadrži i vidljiv betonski stup. Ovi koraci se ponavljaju za sve slike te se u konačnici dobiju četiri oblaka točaka koji sadrže dovoljno informacija za rekonstrukciju cijelog stabla, bez pozadinske smetnje, kao što su trava i grane drugog drveća.

(a) **(b)** **(c)** **(d)**

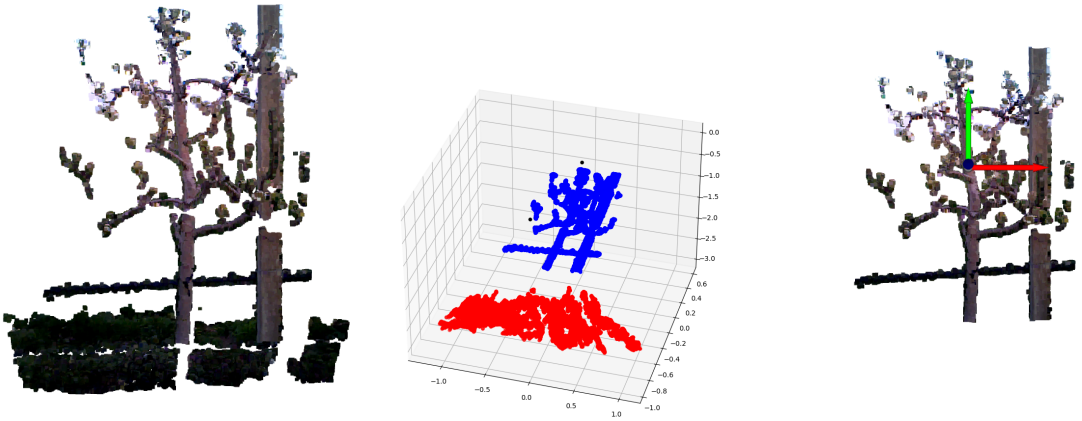
Slika 4.9: *Odabrane RGB slike korištene za rekonstrukciju 15. stabla prije orezivanja.*



(a) **(b)** **(c)** **(d)**

Slika 4.10: *Oblaci točaka dobivenih iz RGB-D slika.*

Kako bi se rekonstruirao model stabla, oblaci točaka istog stabla spajaju se u jedan konačni oblak točaka. U tu svrhu, kao i za potrebe korištenih ugrađenih funkcija iz Open3D biblioteke, prvi oblak točaka se označava kao cilj (engl. *target point cloud*), u daljnjem tekstu *target*, i on je stacionaran, a drugi se označava kao izvor (engl. *source point cloud*), u daljnjem tekstu *source*, koji se rotira i prilagođava targetu. U idućoj iteraciji se spojeni oblak točaka označava kao target, a treći kao source. Sukladno tome, u zadnjoj je iteraciji do sad spojeni oblak točaka target, a zadnji oblak točaka source. U svakoj iteraciji oba oblaka točaka se prvo prethodno obrađuju korištenjem metode *Voxel downsampling* [13]. Voxel predstavlja najmanji dio trodimenzionalnog prostora, tj. 3D piksel. Ova metoda koristi pravilnu mrežu kako bi uniformno generirala smanjeni oblak točaka iz ulaznog oblaka točaka. Ovaj korak uvelike



(a) *Inicijalni oblak točaka.* **(b)** *Segmentacija trave.* **(c)** *Oblak točaka nakon uklanjanja trave.*

Slika 4.11: *Proces uklanjanja trave iz oblaka točaka pomoću K-means algoritma.*

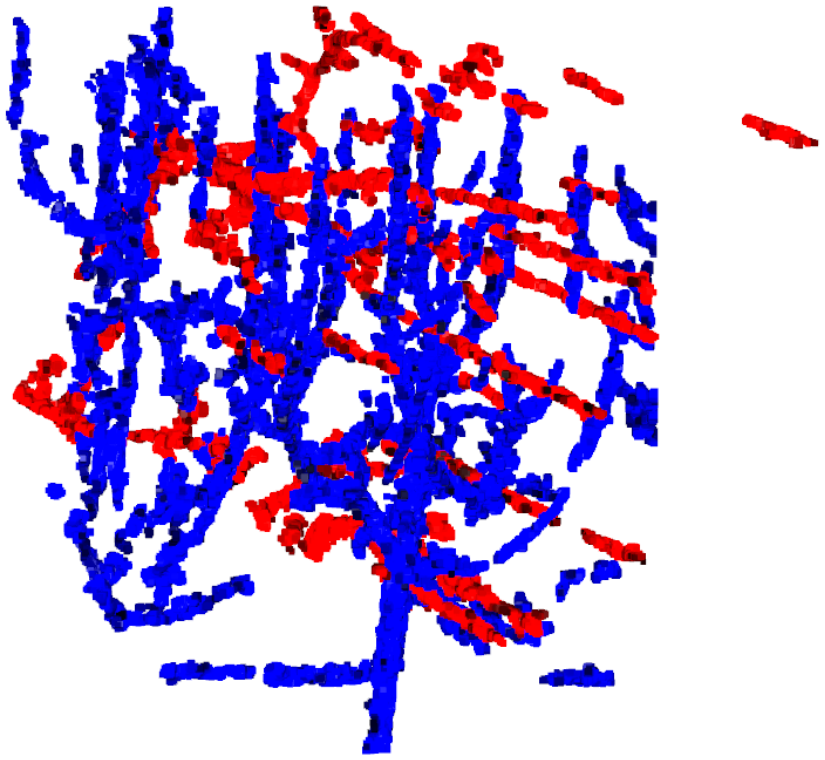
smanjuje broj točaka, ali zadržava kvalitetu podataka, tj. stablo ne gubi oblik te su grane i dalje jasno vidljive što čini algoritam efikasnijim.

Idući korak podrazumijeva globalnu registraciju, za koju nije potrebno inicijalno poudaranje oblaka točaka te se ovakva metoda koristi za lokalne metode, kao što je ICP algoritam [14]. Ova metoda koristi RANSAC (engl. *Random sample consensus*) metodu za estimiranje parametara. U svakoj iteraciji, nasumične točke su odabrane iz sourca te njihove korespondentne točke iz targeta se identificiraju upitom najbližih susjeda u 33-dimenzionalnom FPFH (engl. *Fast Point Feature Histograms*) prostoru značajki [14]. Korak izbacivanja lažnih poudaranja koristi efikasne algoritme za njihovo brzo uklanjanje u ranoj fazi algoritma. Samo točke koje prođu korak izbacivanja se koriste za računanje transformacije, koja se onda validira na cijelom oblaku točaka. RANSAC parametri se određuju na temelju empirijskih vrijednosti koje rezultiraju iz promatranja rezultata na cijelom podatkovnom skupu. Za daljnje podešavanje poravnavanja, koristi se ICP algoritam, čiji ulaz predstavlja rezultat globalne registracije. Inicijalno, početni broj ponavljanja je postavljen na 60 kako bi se odredio potreban broj iteracija. Eksperiment je proveden na prvih 30% podatkovnog skupa i praćen je redni broj iteracije na kojoj se pronašlo najbolje preklapanje. Ovim eksperimentom je utvrđeno da je potrebno najviše pet iteracija da se pronađe preklapanje, ako ono postoji. Najbolje preklapanje se određuje pomoću dva kriterija: broj preklopljenih točaka i srednja kvadratna pogreška (engl. *Root mean squared error RMSE*) svih korespondencija parova točaka.

Zbog situacija u kojima sve iteracije promatranog oblaka točaka dovode do visokih RMSE vrijednosti zbog lošeg poravnavanja ICP algoritma i zbog situacija u kojima se preklapa nešto veći broj točaka (do 20), ali s lošijom RMSE vrijednosti u usporedbi s prethodno izračunatim preklapanjem, predlažu se dvije referentne RMSE vrijednosti, *initialBest_rmse* i *currentBest_rmse*, koje se određuju za svako stablo pojedinačno.

Da bi se odredila vrijednost koja ukazuje na loše preklapanje oblaka točaka, test je obavljen za prvih 15 stabala. Svaka iteracija je ručno i vizualno pregledana i kategorizirana kao dobra ili loša usklađenost i zabilježene su njihove RMSE vrijednosti. Ovaj test je pokazao da RMSE vrijednosti za sve daljnje registracije ne odstupaju više od 50% od početne vrijednosti RMSE ako je početno poravnanje iz globalne registracije vizualno dobro usklađeno. Stoga je početna referentna vrijednost, koja se naziva *initialBest_rmse*, izvedena iz preklapanja prva dva

oblaka točaka unutar prvih pet iteracija. Ova vrijednost služi kao prag za sljedeće iteracije. Ako su u sljedećih pet iteracija za druge oblake točaka sve dobivene RMSE vrijednosti, u daljnjem teksu zvane *inlier_rmse*, znatno više (što ukazuje na vizualno neispravna preklapanja) i ako trenutna najbolja *inlier_rmse* vrijednost prelazi 50% od *initialBest_rmse* vrijednosti, preklapanje se smatra netočnim. Ovaj je kriterij uveden kako bi se izbjegle neusklađenosti koje su vizualno neispravne, bez ručnog pregledavanja svake iteracije. Primjer neispravnog preklapanja vidljiv je na slici 4.12.

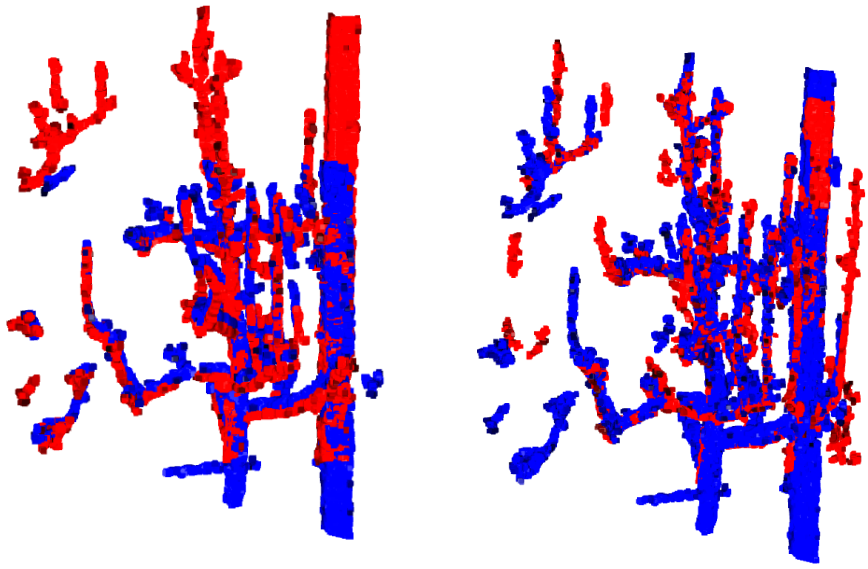


Slika 4.12: *Primjer vizualno neispravnog preklapanja. Source oblak točaka (crvene točke) je krivo rotiran.*

Druga referentna vrijednost, koja se naziva *currentBest_rmse*, koristi se kada se dogodi situacija s manjim brojem točaka preklapanja, ali s višim *inlier_rmse*. U nekim slučajevima to bi moglo dovesti do toga da RMSE vrijednost bude puno viša i tako dati loš rezultat. Kako bi se spriječili takvi slučajevi, za svaku iteraciju također se provjerava je li vrijednost inlier_rmse veća od 20% vrijednosti *currentBest_rmse*, kako bi i RMSE vrijednost bila što niža i kako bi se dobio što veći broj korespondentnih točaka. Prag od 20% određen je empirijski, isprobavanjem različitih postotaka i pronalaženjem najboljeg omjera između velikog broja korespondentnih točaka i niske vrijednosti RMSE.

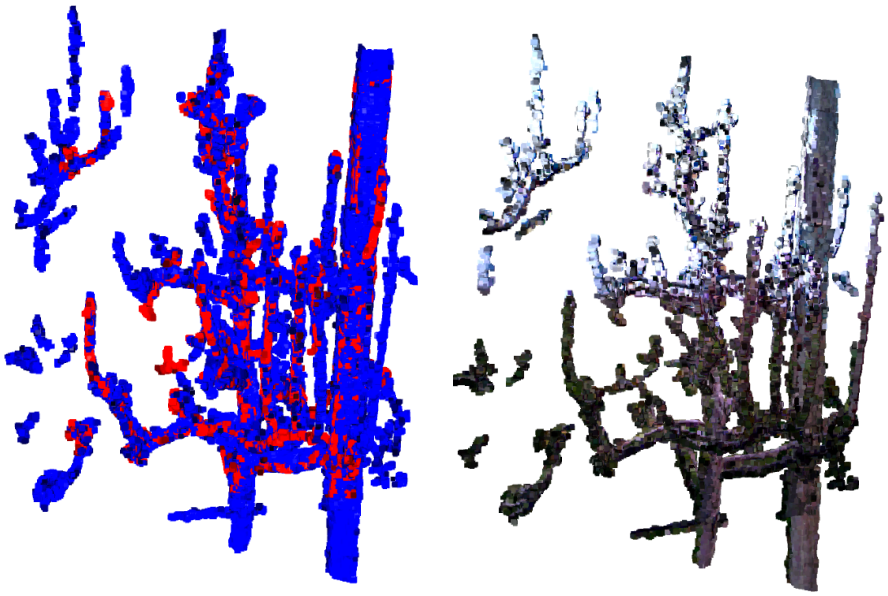
Osim toga, na kraju se provodi vizualni pregled kako bi se identificirale potpuno netočne rekonstrukcije gdje je najbolja RMSE vrijednost preklapanja prevelika. Koristeći ovu metodu, svi modeli stabala u skupu podataka rekonstruirani su i prije i nakon rezidbe te je postojeći podatkovni skup proširen s rekonstruiranim modelima svakog stabla. Primjer jednog rekonstruiranog stabla po koracima vidljiv je na slici 4.13. Source točke su obojane crvenom, a target plavom bojom. Na nekim oblacima točaka je vidljiv i betonski stup, kao što je vidljiv i na slici

4.13, no on je nepromjenjiv u slikama prije i poslije rezidbe. Stoga ga algoritam neće prepoznati kao granu za rezanje.



(a) *Spajanje prvog i drugog oblaka točaka.*

(b) *Spajanje s trećim oblakom točaka.*



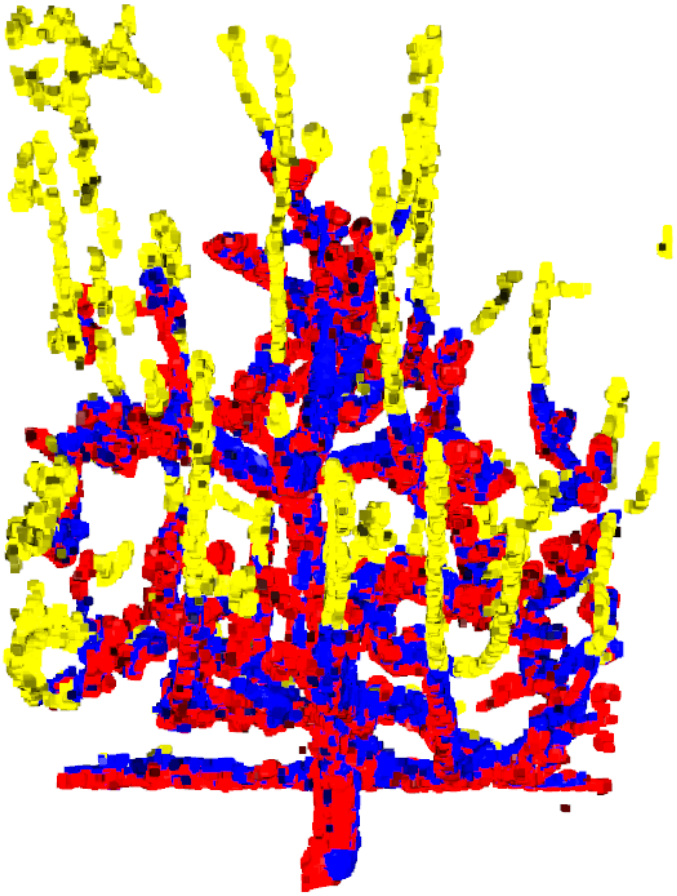
(c) *Spajanje sa četvrtim oblakom točaka.*

(d) *Konačni spojeni i obojeni oblak točaka.*

Slika 4.13: *Primjer koraka rekonstrukcije stabla.*

4.2. Označavanje grana za rezidbu

Nakon što se provede gore navedeni algoritam na svim stablima i stvore se rekonstruirani modeli prije i poslije rezidbe, potrebno je označiti one grane koje su orezane. Te označene grane koje su orezane predstavljaju stvarnu vrijednost (engl. *ground truth*) za potrebe učenje neuronske mreže. Označavanje grana se također može postići koristeći navedeni algoritam, ali se za target oblak točaka uzme rekonstruirani model prije rezidbe, a za source rekonstruirani model nakon rezidbe. Nakon preklapanja ova dva modela istim postupkom, dobije se rezultirajući model na kojem na kojem razlika preklapanja predstavlja orezane grane. Označavanje tih grana se provodi tako da se za svaku točku u target oblaku točaka računa udaljenost do najbliže korespondentne točke u sourcu. Ako ta udaljenost prelazi prag od 3 cm, ta točka se smatra točkom koja pripada orezanoj grani. Prag udaljenosti od 3 cm je određen pomoću vizualne inspekcije rezultirajućeg oblaka točaka i predstavlja prihvatljivu granicu pogreške za stabla visine između 150 cm i 200 cm i čije su grane duljine između 30 cm i 100 cm. Prikaz jednog takvog oblaka točaka vidljiv je na slici 4.14.



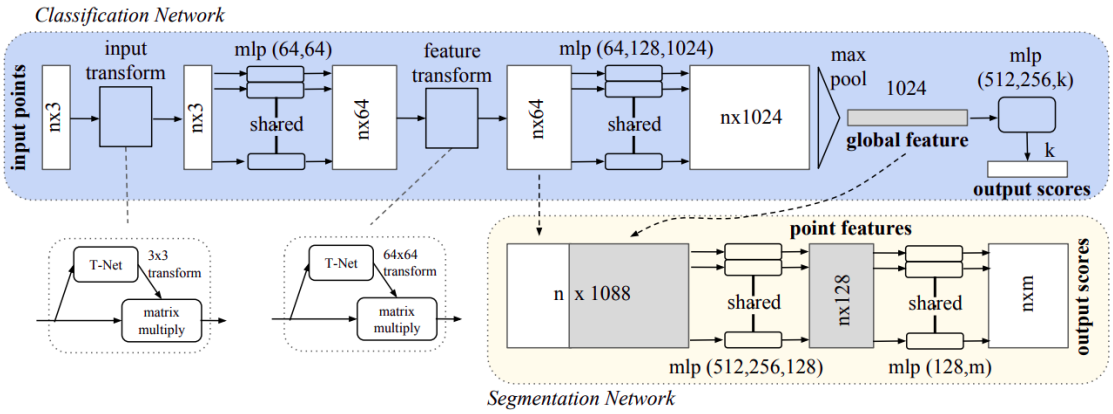
Slika 4.14: *Spojeni modeli oblaka točaka prije (plavo) i poslije (crveno) rezidbe. Žute točke predstavljaju orezane grane.*

Ovakvi spojeni modeli oblaka točaka, koji su na ovaj način obojani (crveno i plavo za neorezane, žuto za orezane točke) su potrebni za učenje neuronske mreže. Oznake (engl. *label*) se učitaju iz modela tako što se stvori niz koji je popunjen nulama i jedinicama, pri čemu indeks

tog niza odgovara indeksu točke u modelu. Nula označava crvene i plave točke, tj. neorezane grane, a jednica označava žute točke, tj. orezane grane. Dakle, ulazi u neuronsku mrežu sastoje se od niza koji sadrži koordinate svake točke te niza koji sadrži odgovarajuće oznake.

4.3. Neuronska mreža za učenje detekcije grana za rezidbu

Neuronska mreža korištena u ovom radu temeljena je na *PointNet++* arhitekturi mreže, predstavljene u radu [15]. *Pointnet++* mreža predstavlja nadogradnju na *PointNet* mrežu, predstavljenu u radu [16]. Njena arhitektura vidljiva je na slici 4.15.



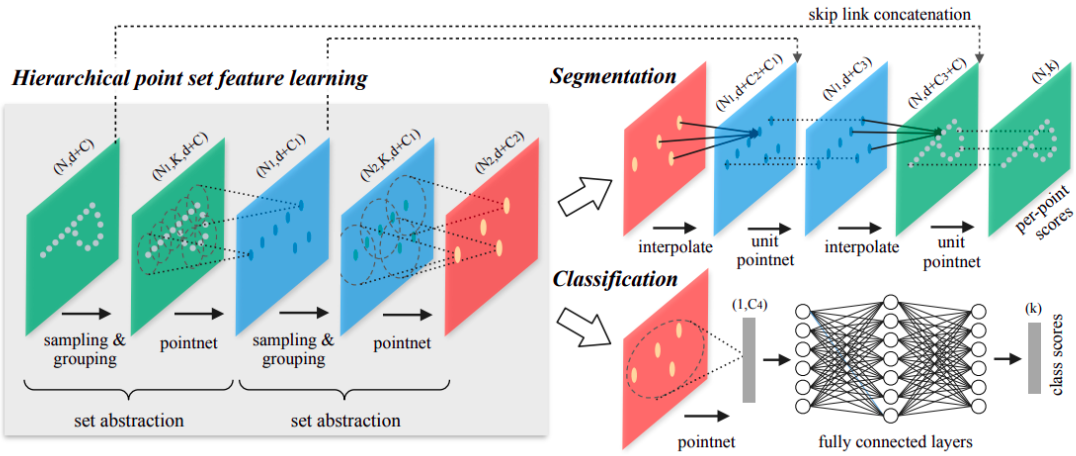
Slika 4.15: *Arhitektura PointNet mreže. [16].*

Klasifikacijska mreža za ulaz prima *nx3* točaka, primjenjuje transformaciju podataka, zatim agregira značajke točaka slojem sažimanja po maksimalnoj vrijednosti (engl. *max-pool layer*). Izlaz su klasifikacije za k klasa. Segmentacijska mreža je samo proširenje na klasifikacijsku. Spaja globalne i lokalne značajke te izračunava oznaku za svaku točku. Ključni moduli ove mreže uključuju: sloj sažimanja po maksimalnoj vrijednosti kao simetrična funkcija za agregaciju informacija svih točaka, kombinacija informacija lokalnih i globalnih točaka i udružena struktura mreža za usklađivanje ulaznih točaka i njihovih značajki. Također ova mreža sadrži i višeslojne perceptron mreže (engl. *Multi-Layered Perceptron, MLP*). Na ovakvu mrežu ne utječu permutacije ulaznog skupa i mreža može proizvoljno aproksimirati bilo koju funkciju kontinuiranog skupa. Međutim, ovoj mreži nedostaje sposobnost uočavanja lokalnog konteksta na većim skalama. Ovaj problem rješava *PointNet++* mreža.

Arhitektura *PointNet++* mreže vidljiva je na slici 4.16.

PointNet++ mreža upotrebljava hijerarhijsku strukturu učenja značajki. Ta hijerarhijska struktura sadržava niz apstrakcijskih razina (engl. *set abstraction layers*). Na svakoj razini se skup točaka obrađuje i apstrahira, kako bi se proizveo novi skup s manje elemenata. Taj niz apstrakcijskih razina sadržava tri ključna sloja: sloj uzorkovanja (engl. *sampling layer*), sloj grupiranja (engl. *grouping layer*) i *PointNet* sloj. Sloj uzorkovanja odabire skup točaka iz ulaznih točaka koje predstavljaju težišta lokalnih regija. Zatim sloj grupiranja konstruira skupove lokalnih regija pronalaženjem susjednih točaka oko težišta. *PointNet* sloj kodira uzorke lokalne regije u vektore značajki.

Česta pojava neujednačenosti gustoća u različitim područjima predstavlja izazov za učenje značajki skupa točaka. Značajke naučene u gustim podacima možda se neće generalizirati na rijetko uzorkovana područja. U područjima niske gustoće nije moguće detaljno pregledati točke i uočavati uzorke, jer takvi uzorci mogu biti iskrivljeni zbog manjka uzorkovanih točaka. U takvim slučajevima potrebno je tražiti uzorke većih razmjera u širem okruženju. Ovaj cilj postignut je s *PointNet* slojevima prilagodljivim gustoći (engl. *Density adaptive PointNet layers*)



Slika 4.16: *Arhitektura PointNet++ mreže. [15].*

koji uče kombinirati značajke iz regija različitih razmjera kada se mijenja gustoća uzorkovanja. Hijerarhijska mreža zajedno s *PointNet* slojevima prilagodljivim gustoći čini *PointNet++* mrežu. Svaka razina apstrakcije u ovoj mreži izvlači više skala lokalnih uzoraka i kombinira ih prema gustoći lokalnih točaka. Grupiranje lokalnih regija i kombiniranje značajki iz različitih skala odrađeno je s pomoću sloja grupiranja s više skala (engl *Multi-scale grouping, MSG*). Ovaj sloj, zajedno s *PointNet* slojevima izvlači uzorke iz više skala i spaja ih, formirajući značajku s više skala (engl. *Multi-scale feature*). Na ovaj način moguće je učiti mrežu i na uzorcima manjih skala, kao što su male promjene u obliku grana i vidljivi pupoljci.

Propagacijska razina slijedi nakon razine apstrakcije i ona sadržava slojeve propagacije značajki točaka (engl. *Point feature propagation for set segmentation*). Prilikom učenja mreže poželjno je imati značajke za sve točke, ali originalni skup točaka se u apstrakcijskoj razini poduzorkuje zbog uštede računalnih resursa. Jedno rješenje je da se svaka točka označi kao težište u razini apstrakcije, no to uvelike povećava potrošnju računalnih resursa. Drugo rješenje je propagiranje značajki s poduzorkovanih točaka na izvorne točke omogućava prijenos informacija iz manjeg skupa točaka nazad na sve izvorne točke. To se postiže interpolacijom, gdje se značajke iz poduzorkovanih točaka šire i prilagođavaju izvornim točkama, čime model zadržava precizne informacije za sve točke, ali uz smanjeni trošak obrade tijekom faza apstrakcije.

Jedan od modela predstavljenih u [15], koji koristi *PointNet++* arhitekturu je model za segmentaciju dijelova objekata na *ShapeNet* modelima [17]. Ovaj model, za potrebe ovog rada, je prilagođen za rad sa samo dvije klase te mogućnost rada i bez predanog niza oznaka za svaku točku. Jedna klasa predstavlja točke koje su označene kao neorezane (u programskom kodu su označene kao klasa "0"), a druga predstavlja točke koje su označene kao orezane (u programskom kodu označene kao klasa "1"). Model sadrži:

- Dvije razine *PointNet* apstrakcije s MSG grupiranjem (*PointNetSetAbstractionMsg*)
- Jedna razina *PointNet* apstrakcije (*PointNetSetAbstraction*)
- Tri propagacijska sloja (*PointNetFeaturePropagation*)
- 1D konvolucijski sloj
- Sloj normalizacije po serijama (engl. *Batch normalization*)
- ReLu aktivacijska funkcija
- Sloj izbacivanja (engl. *dropout*)

- 1D konvolucijski sloj
- Logaritamska softmax funkcija

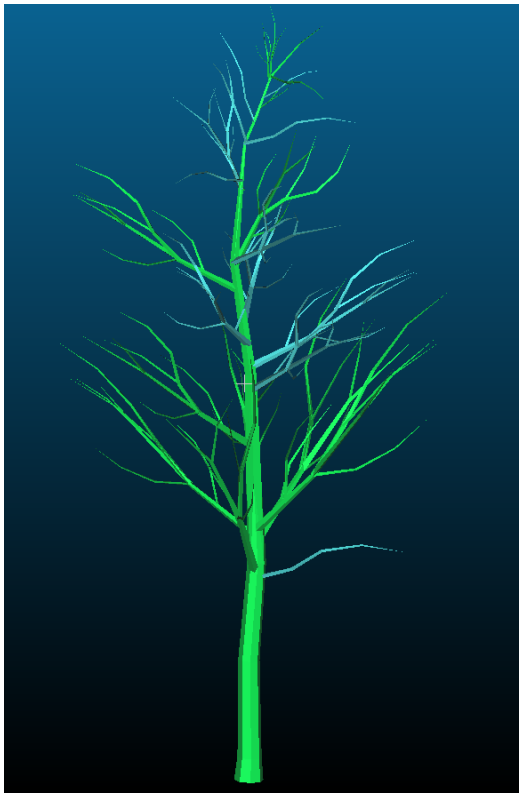
Kao zadnji sloj korištena je logaritamska softmax funkcija zbog jednostavnosti implementacije funkcije gubitka (engl. *loss function*). Za funkciju gubitka koristi se funkcija negativne logaritamske vjerodostojnosti. U suštini, izlaz iz neuronske mreže je niz s dva elementa. Svaki od elemenata predstavlja vjerojatnost pripadanju određenoj klasi za tu točku. Za odabir klase, uzima se veća vjerojatnost, a s obzirom na to da je zbroj tih vjerojatnosti uvijek 1, na ovaj način se oponaša sigmoid funkcija. Svi hiperparamteri neuronske mreže su uobičajene vrijednosti (engl. *default values*). Cijela implementacija mreže napravljena je u PyTorch biblioteci.

5. EVALUACIJA

U ovom poglavlju opisana je evaluacija algoritma za rekonstrukciju 3D modela stabla i prikazani su njezini rezultati. Evaluacija algoritma je provedena na sintetičkom modelu, tako da se na dobivene oblake točaka dodaje umjetni šum te se provede cijeli algoritam rekonstrukcije. Nakon izvedbe cijelog algoritma, uspoređuje se izvorni oblak točaka sintetičkog modela i rekonstruiranog, a s obzirom da je poznata geometrija sintetičkog modela, tj. postoji referentna vrijednost (engl. *ground truth*), moguće je odrediti točnost preklapanja metrikom presjek preko unije (engl. *intersection over union*). Ako se utvrdi visoka točnost rekonstrukcije na sintetičkom modelu, očekuje se da će sa sličnom točnošću raditi i na slikama podatkovnog skupa.

5.1. Rekonstrukcija sintetičkog modela

S obzirom na to da je algoritam vremenski intenzivan i teško je procijeniti točnost rekonstrukcije 3D modela stabla jer ne postoji stvarna vrijednost (engl. *ground truth*) na kojoj bi se temeljila procjena, najprije je provedena evaluacija na sintetičkom modelu. Ovakav pristup naziva se osnovnom provjerom valjanosti i služi za postavljanje temeljne referentne vrijednosti izvedbe, osiguravajući da su temeljne funkcionalnosti, kao što su pravilno poravnavanje oblaka točaka globalnom registracijom i dodatno usklađivanje ICP algoritmom, provjerene prije primjene algoritma na složenije podatke. Za potrebe ovog pristupa korišten je jednostavan 3D model stabla, vidljiv na slici 5.17. Taj 3D model stabla je skaliran tako da mu visina bude 200 cm. Odabran je taj model zato što nalikuje stablima po njegovoj visini, debljini debla i rasporedu, broju i duljini grana u pravom podatkovnom skupu.



Slika 5.17: *Sintetički 3D model korišten za osnovnu provjeru valjanosti.*

Model je postavljen na scenu pomoću klase Open3D Visualizer [18]. Nadalje, također je postavljen pinhole camera objekt na scenu, definiran intrinzičnim parametrima kamere korištene

za prikupljanje stvarnog podatkovnog skupa. S obzirom na to da su potrebno samo četiri RGB i dubinske slike, kamera je postavljena u 4 položaja u kojima su snimljene slike: dvije sprijeda te po jedna sa svake bočne strane stabla. Snimanje tih slika je napravljeno na isti način kao što bi i u pravom voćnjaku, tako da su na slikama vidljive sve grane promatranog stabla.

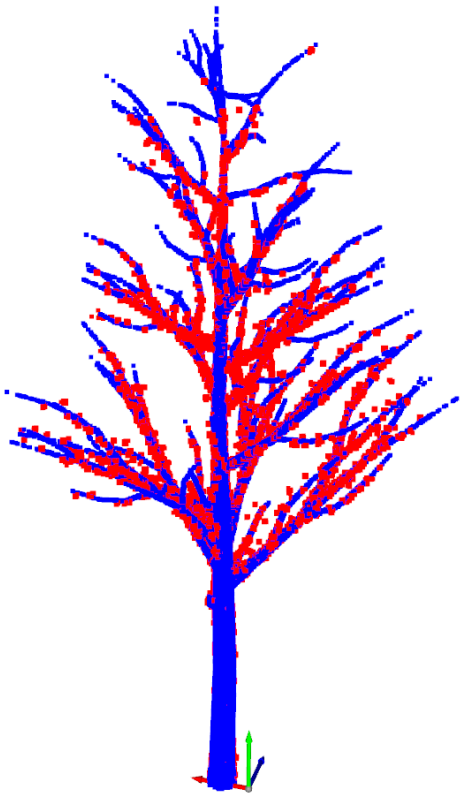
Kako je na stvarnim slikama prisutan šum zbog loše kvalitete kamere i prisutnosti pozadine (npr. trave), a rezultirajuća slika sintetičkog 3D modela stabla nema nikakvih gubitaka niti smetnji, algoritam na takvoj slici ima točnost od 100%. Zbog toga je dodan umjetni šum na slike. Taj šum je dodan na temelju modela predstavljenog u [19] te preuzetog s web stranice [20]. Model na postojuće dubinske slike dodaje umjetan šum tako da simulira šum na slikama uslikanim Kinect senzorom. Male promjene su napravljene u navedenom modelu dodavanja umjetnog šuma kako bi dubinske slike bile što sličnije onima prikupljenim u podatkovnom skupu. Navedene promjene u modelu su određene emprijiski, na način da su se hiperparametri mijenjali i vizualnom usporedbom dobivene slike sa dubinskom slikom jednog stabla iz podatkovnog skupa. Primjer dubinske slike sintetičkog 3D modela i iste te dubinske slike zašumljenje navedenim modelom vidljivi su na slikama 5.17 i 5.19.

Slika 5.18: *Dubinska slika sintetičkog 3D modela.*

Slika 5.19: *Dubinska slika sa dodanim umjetnim šumom.*

Zatim je primijenjen algoritam rekonstrukcije, opisan u 4. poglavlju. Budući da rekonstruirani oblak točaka i sintetički model nisu međusobno usklađeni, tj. model je u obliku mreže trokuta (engl. *mesh*) i ima drugačije koordinate kada se i model i oblak točaka postave na scenu, potrebno je uzorkovati oblak točaka iz sintetičkog modela i poravnati ih pomoću algoritma poravnavanja korištenog unutar algoritma rekonstrukcije. Kako bi se to postiglo, korišten je CloudCompare, softver za obradu 3D oblaka točaka i mreža trokuta [21]. Nakon uzorkovanja, 3D sintetički model ima jednak broj točaka kao i rekonstruirani oblak točaka. Rezultat preklapanja oblaka točaka vidljiv je na slici 5.20.

Za procjenu točnosti algoritma za rekonstrukciju korištene su dvije metrike: presjek preko unije (engl. *intersection over union*), u nastavku IoU, i *chamfer distance*. IoU često se koristi s plohama i volumenima, pa je u ovom slučaju stvoren umjetni volumen oko svake točke u sintetičkom oblaku točaka i provjerava se nalazi li se odgovarajuća točka iz rekonstruiranog oblaka točaka unutar tog volumena. Volumen je sferičan, a vrijednosti praga predstavljaju njegov polumjer. S obzirom na prethodnu definiciju, za vrijednost praga od 1 cm, 63.39% točaka rekonstruiranog modela nalazi se unutar volumena. Ako se prag proširi na 5 cm, sadržano je 94.66% točaka. Pragovi od 1 cm i 5 cm predstavljaju prihvatljivu granicu pogreške od 0.5% i 2.5% stabla visine 200 cm. Druga metrika koja se često koristi za procjenu sličnosti između dva skupa točaka, A i B, je *chamfer distance*. *Chamfer distance* definirana je formulom 5-1.



Slika 5.20: *Preklopljeni sintetički model oblaka točaka (plave točke) i rekonstruirani sintetički model (crvene točke).*

$$\sum_{i=1}^{n} d(A_i, B) + \sum_{i=1}^{n} d(B_i, A) \tag{5-1}$$

gdje je:

- n - broj točaka u oblaku točaka
- $d(A_i, B)$ - udaljenost i-te točke iz oblaka točaka A do njenog najbližeg susjeda u oblaku točaka B
- $d(B_i, A)$ - udaljenost i-te točke iz oblaka točaka B do njenog najbližeg susjeda u oblaku točaka A

Drugim riječima, *chamfer distance* predstavlja prosječnu udaljenost između točaka. Izračunati *chamfer distance* za oblak točaka dobiven uzorkovanjem sintetičkog modela i rekonstruiranim oblakom točaka je 2.897 cm. Ova udaljenost također uključuje točke koje su stršeće vrijednosti i nisu spojene sa promatranim stablom. Ako se te točke zanemare pri izračunavanju ove metrike, ona pada na 2.2 cm.

5.2. Evaluacija na podatkovnom skupu BRANCH

S obzirom da je provođenje algoritma opisanog u 4. poglavlju vremenski zahtjevan proces, od ukupno 184 uslikanih stabla, algoritam je u svrhu ovog diplomskog rada proveden na 70 stabala kruške. Uspješnost algoritma iznosi 74.29%, što znači da je uspješno rekonstruirano 52 modela

stabla prije i poslije rezidbe. 18 neuspješnih rekonstrukcija uzrokovano je jednim od sljedeća tri glavna problema:

1. ICP algoritam ne može pronaći preklapanje ako *source* oblak točaka ne sadrži deblo stabla, već samo vrhove manjih grana.

2. Ako rekonstrukcija stabla iz prva dva oblaka točaka daje visoke RMSE vrijednosti tada ne postoji kriterij za identifikaciju netočne rekonstrukcije, osim vizualne inspekcije. Ovaj problem se najčešće događa kada je početno preklapanje takvo da je jedan oblak točaka krivo rotiran, kao što je vidljivo na slici 4.12.

3. Ako je *inliner_rmse* vrijednost za rekonstrukciju prva dva oblaka točaka u gornjim granicama prihvatljivosti, ali i da je i dalje vizualno ispravno, postoji šansa da iduća preklapanja budu neispravna zbog krive rotacije oblaka točaka. Zbog praga koji je postavljen na 50% vrijednosti najboljeg početnog preklapanja, moguće je da i netočno preklapanje zadovoljava taj uvjet i algoritam smatra točnim preklapanjem. Empirijski je utvrđeno da su najčešće RMSE vrijednosti u intrevalu od $3.2 \cdot 10^{-2}$ do $3.8 \cdot 10^{-2}$.

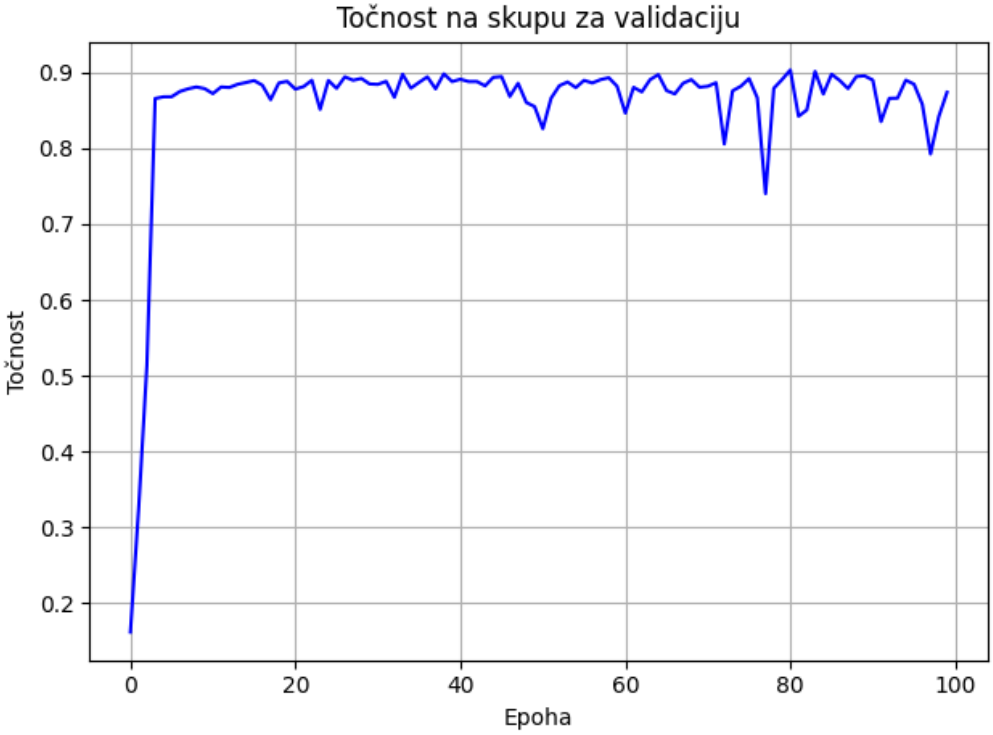
Primjer uspješno rekonstruiranog stabla vidljiv je na slici 4.13, a primjer dva preklopljena modela s označenim granama vidljiv je na slici 4.14. S obzirom na potrebu za ručnim odabirom i pregledom određenih dijelova, kao i to da algoritam zahtijeva značajne računalne resurse, izvođenje ovog algoritma je vremenski zahtjevan posao. Unatoč tome, 52 uspješno rekonstruiranih modela stabla predstavlja podatkovni skup na kojem je neuronska mreža, opisana u poglavlju 4., trenirana i testirana.

5.3. Evaluacija rezultata dobivenih pomoću neuronske mreže

Modeli drveća prije rezidbe predstavljeni su točkama u 3D prostoru te je svakoj točki dodijeljena oznaka koja govori pripada li točka orezanoj grani ili ne. Oznake su dodijeljene na način opisan u potpoglavlju 4.2. Niz koji sadrži koordinate svih točaka modela i niz koji sadrži oznake za svaku tu točku predstavljaju ulaze za učenje neuronske mreže. Izlaz je predviđeni niz oznaka za svaku točku promatranog modela. Evaluacija je provedena kroz četiri metrike: preciznost (engl. *precision*), točnost (engl. *accuracy*), odziv (engl. *recall*) i F1 rezultat (engl. *F1 score*). Ove četiri metrike su najčešće metrike za klasifikacijske modele.

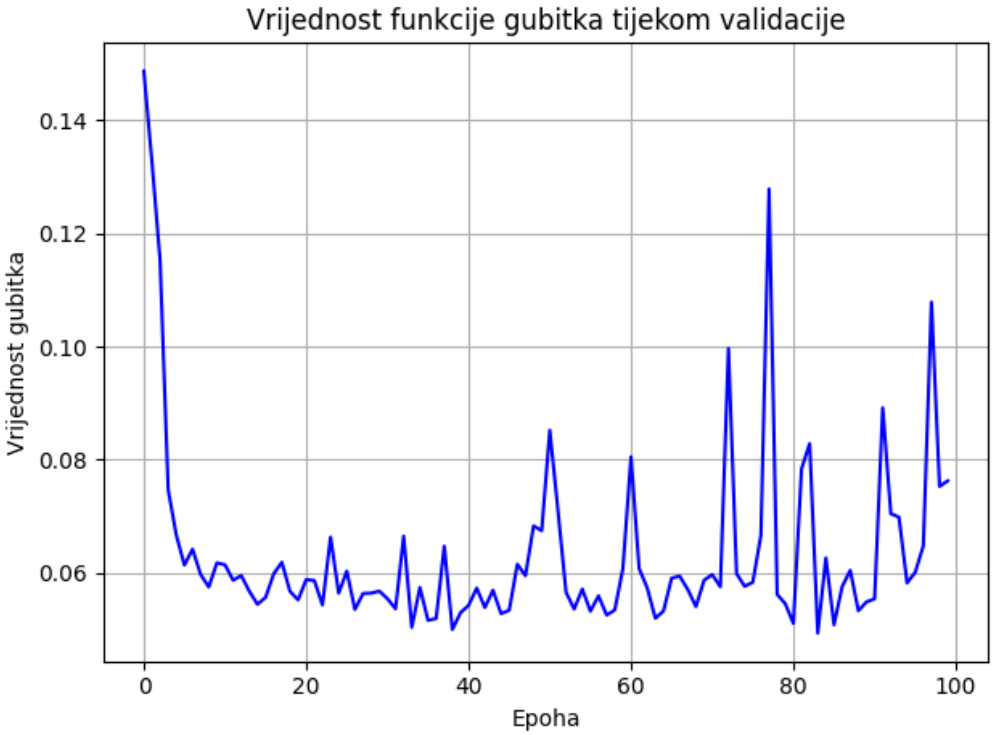
Prije početka treniranja mreže, podatkovni skup je podijeljen na tri dijela: skup za treniranje, skup za testiranje i skup za validaciju, u postotnim omjerima 70:15:15, što daje 36 modela za treniranje, 8 za testiranje i 8 za validaciju. Treniranje je obavljeno na osobnom računalu, čija grafička kartica ima 4 GB VRAM memorije. Zbog ograničene količine VRAM memorije, učenje je bilo moguće provesti na 10 000 točaka svakog modela, a svaki model ima između 500 000 i 900 000 točaka. Točke koje se odabiru za treniranje, testiranje i validaciju su nasumične i odabiru se za svaki model pojedinačno. Ipak, zbog male količine podataka, vrijeme treniranja je 22 minute kroz 100 epoha. Najvišu točnost na validacijskom skupu mreža je postigla u 80. epohi i ona iznosi 90.312%, no kroz 5 do 10 epoha već postiže pribilžnu točnost od 86%, što je vidljivo na slici 5.22. Točnost na trening skupu konsantno raste, što je vidljivo na slici 5.21. Ovo sugerira na pojavu pretjeranog usklađivanja (engl. *overfitting*) na trening skupu podataka. Ova pojava se mogla i očekivati s obzirom na mali broj podataka u skupu i velikim brojem epoha. Ovo je također vidljivo i na grafovima funkcijama gubitka, što je vidljivo na slikama 5.23 i 5.24. Zaključak je da se treniranje može zaustaviti nakon 10 epoha.

Slika 5.21: *Graf točnosti na trening skupu kroz epohe.*



Slika 5.22: *Graf točnosti na validacijskom skupu kroz epohe.*

Slika 5.23: *Graf vrijednosti funkcije gubitka tijekom treniranja.*



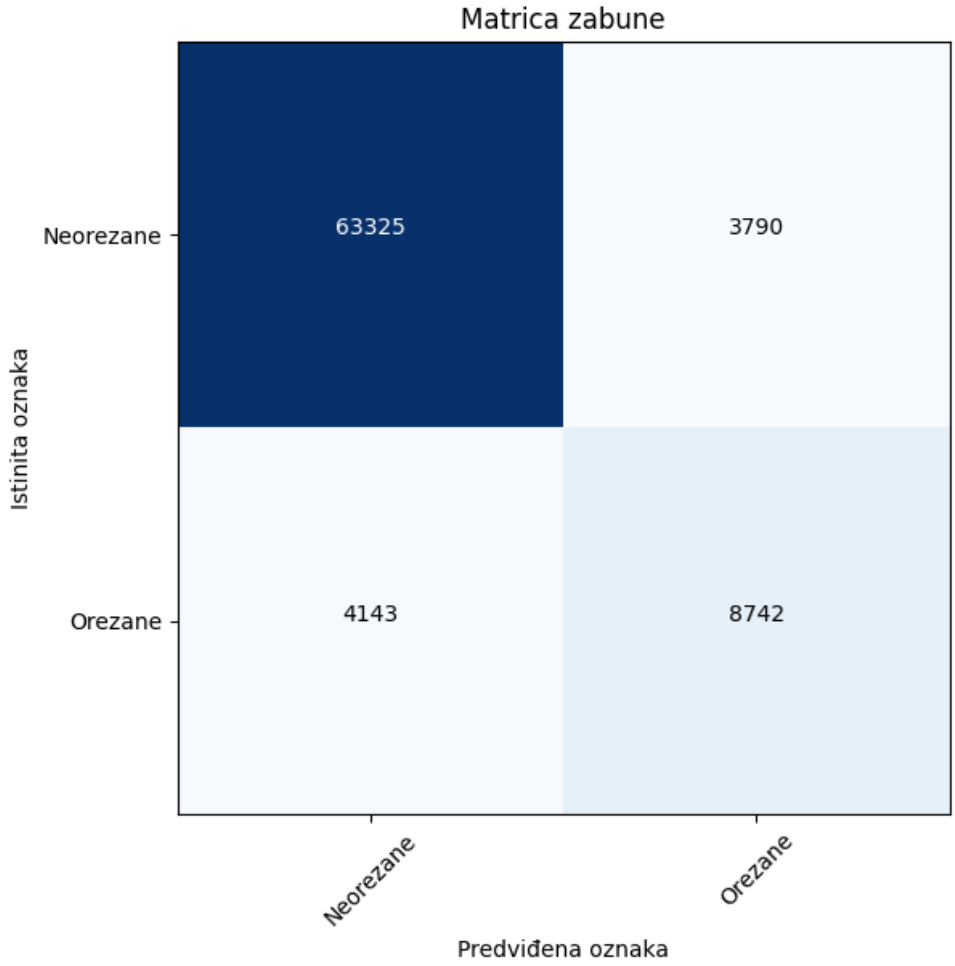
Slika 5.24: *Graf vrijednosti funkcije gubitka tijekom validacije.*

Nakon treniranja, neuronska mreža je testirana na skupu za testiranje. Metrike su izračunate na cijelom skupu, uzimajući u obzir sve točke ulaznog skupa i sve oznake u izlaznom skupu. Dobivene metrike su prikazane u tablici 5.2. Sve metrike osim točnosti su

izračunate za klasu orezanih točaka. Matrica zabune za sve točke testnog skupa prikazana je na slici 5.25.

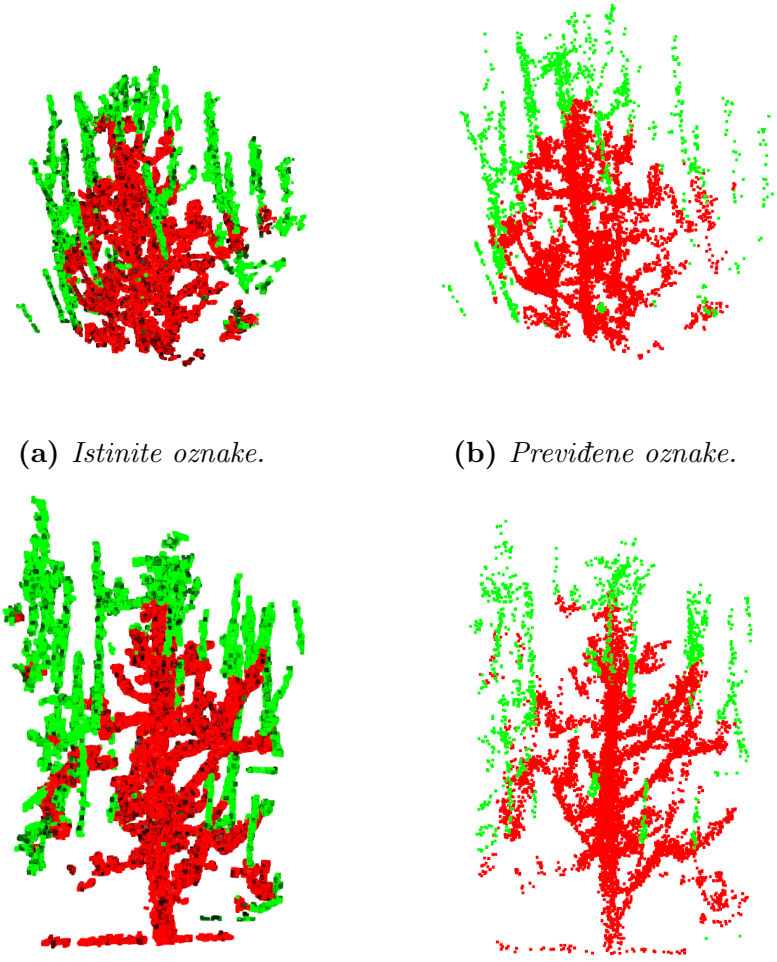
Točnost	Preciznost	Odziv	F1 rezultat
90.08%	69.76%	67.85%	68.79%

Tablica 5.2: *Metrike izračunate na skupu za testiranje.*



Slika 5.25: *Matrica zabune.*

Nekoliko primjera primjene izlaza iz neuronske mreže na testne oblake točaka neorezanog drveća, kao i njihova istinita vrijednost, vidljivi su na slici 5.26, pri čemu su zeleno označene orezane grane, a crveno ostatak drveta.



(a) *Istinite oznake.* **(b)** *Previđene oznake.*

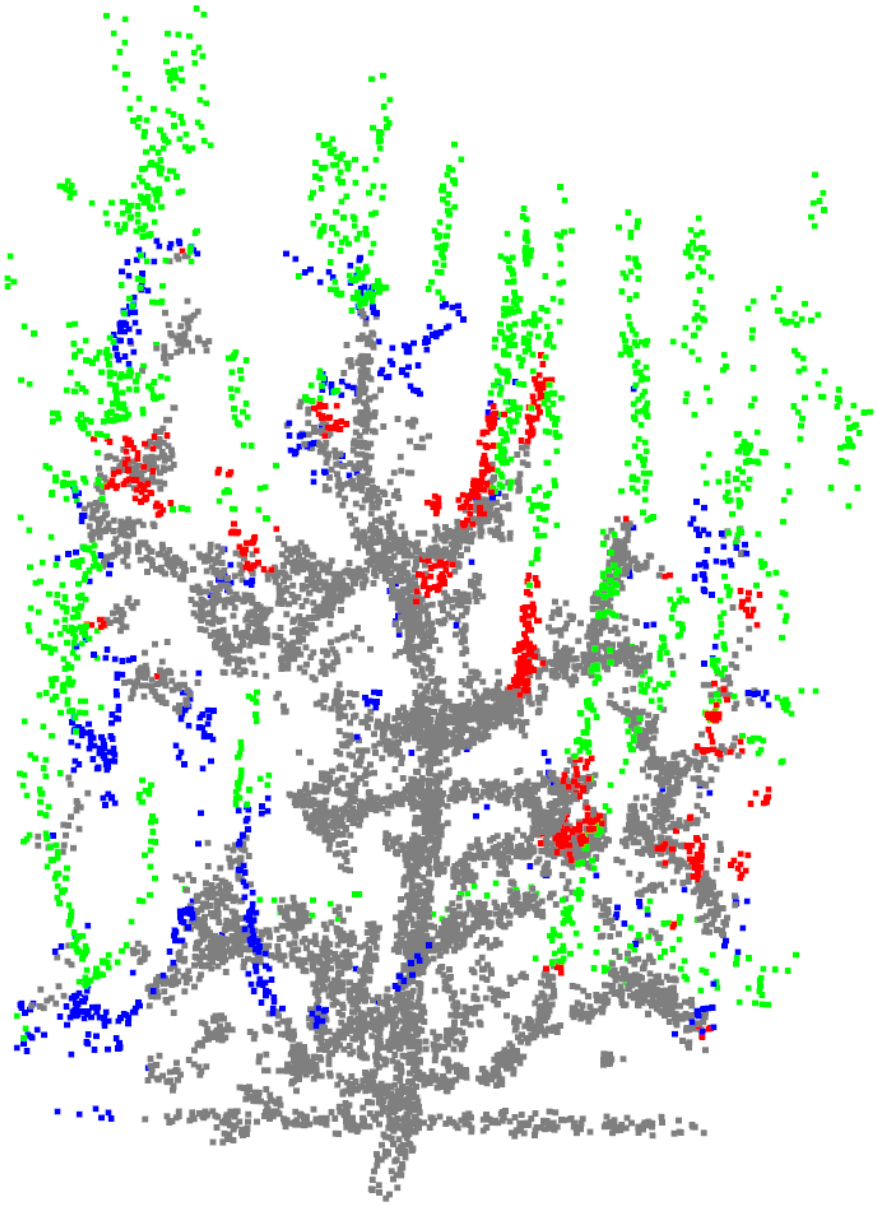
(c) *Istinite oznake.* **(d)** *Previđene oznake.*

Slika 5.26: *Primjer primjene izlaza iz neuronske mreže na oblak točaka.*

Kako bi se vizualno procijenila uspješnost klasifikacije, na slici 5.27 je prikazan primjer jednog oblaka točaka drveta iz testnog skupa. Na njemu su označene:

- Istinito pozitivine (engl. *True positive, TP*) točke u zelenoj boji (orezane grane)
- Istinito negativne (engl. *True negative, TN*) točke u sivoj boji (neorezane grane)
- Lažno pozitivne (engl. *False positive, FP*) točke u crvenoj boji (lažno orezane grane)
- Lažno negativne (engl. *False negative, FN*) točke u plavoj boji (lažno neorezane grane)

Crvene točke, koje predstavljaju lažno pozitivne, su za primjenu u rezidbi važne i njih treba najviše minimizirati. Model ne bi trebao označavati pogrešne grane za rezidbu, jer se takve pogreške ne mogu naknadno ručno popraviti. Također na toj slici je vidljivo da su rezišta nekih orezanih grana označena malo niže od njihovih pravih točaka rezišta. To ukazuje na to da treba dodati dodatni kriterij za odabir točke rezišta. Specifično za taj primjer, u tablici 5.3 su prikazane izračunate metrike, od kojih je preciznost, koja minimizira lažno pozitivne točke, najvažnija i težnja je da bude što veća.



Slika 5.27: *Testni primjer oblaka točaka neorezanog drveta s označenim TP (zelenim), TN (sivim), FP (crvenim) i FN (plavim) točkama.*

Preciznost	Odziv	F1 rezultat
74.99%	70.91%	72.89%

Tablica 5.3: *Metrike izračunate na primjeru sa slike 5.27*

6. ZAKLJUčAK

U ovom diplomskom radu prikupljen je podatkovni skup, BRANCH, RGB-D slika voćaka kruške prije i poslije rezidbe grana. Također, predstavljeni su postupak za rekonstrukciju modela stabla na temelju RGB-D slika i neuronska mreža za detekciju grana za rezidbu. Za prikupljanje podatkovnog skupa korištena je ASUS Xtion PRO live kamera i Robotski operacijski sustav. Algoritam za rekonstrukciju stabla i detekciju točaka koje pripadaju orezanim granama napisan je u programskom jeziku Python te je primijenjen na podatkovnom skupu BRANCH. Također stvoren je model neuronske mreže, temeljen na *PointNet++* arhitekturi. Ova neuronska mreža je istrenirana na prikupljenom i obrađenom BRANCH podatkovnom skupu.

Kako ne postoje cjeloviti modeli uslikanih stabala, provedena je evaluacija algoritma za rekonstrukciju 3D modela na sintetičkom modelu. Za evaluaciju algoritma na sintetičkom modelu, korištene su metrike presjek preko unije (IoU) i *chamfer distance*. Vrijednost IoU za prag od 5 cm iznosi 94.66%, što zapravo govori da je 94.66% točaka smješteno unutar 5 cm njezinog pravog položaja u prostoru. *Chamfer distance* zapravo predstavlja prosječnu udaljenost između točaka iz dva različita skupa, i ono iznosi 2.897 cm, a ako se izbace stršeće vrijednosti koje nisu spojene sa promatranim stablom, ona pada na 2.2 cm. Ova metrika također govori da su točke u prostoru smještene vrlo blizu, uzimajući u obzir prosječne veličine uslikanih stabla. S ovim rezultatima je osigurano da algoritam, ako napravi uspješnu rekonstrukciju, položaji točaka su približne njihovim stvarnim vrijednostima.

Algoritam za rekonstrukciju vremenski je zahtjevan za izvođenje, a na stvarnim modelima ima uspješnost rekonstrukcije od 74.29%. Uspješno rekonstruirani modeli su oni za koje je algoritam proveden do kraja, a neuspješni su oni za koje je algoritam tijekom izvođenja zaustavljen i rekonstrukcija nije uspjela. Zbog vremenske složenosti, u ovom diplomskom radu rekonstruirano je 70 stabla, od ukupno 184 snimljena stabla u BRANCH podatkovnom skupu. Usporedbom rekonstruiranih stabala prije i poslije rezidbe, označene su točke koje pripadaju orezanim granama. Ove točke, zajedno s modelima prije rezidbe čine ulaze za učenje neuronske mreže.

Na testnom skupu, neuronska mreža je postigla točnost 90.08%, preciznost 69.76%, odziv 67.85% i F1 rezultat 68.79%. Analizom i vizualizacijom rezultata, može se zaključiti da neuronska mreža ima veliku točnost, ali da se trebaju minimizirati pojave lažno pozitivnih primjera, kako bi preciznost bila što veća tj. da neuronska mreža ne označava grane za rezanje koje ne treba orezivati. Važno je imati na umu da je mreža trenirana na vrlo malo podataka te da je unatoč tome postignuta visoka točnost, dok za ostale metrike ima prostora za poboljšanje.

Najveći utjecaj na dobivene rezultate rekonstrukcije imaju šum kamere uzrokovan osjetljivošću senzora na sunčevu svjetlost i pozadinski šum, točnije trava i grane drugih stabla. Najveći nedostatak prilikom treniranja mreže je relativno mali broj ulaznih podataka. Navedene probleme je moguće riješiti korištenjem senzora manje osjetljivih na sunčevu svjetlost prilikom prikupljanja podatkovnog skupa. Također moguće je i snimiti video u kojem bi cijelo stablo bilo vidljivo iz više kutova, pa na taj način prikupiti dovoljno informacija za cjelokupnu rekonstrukciju modela. Uz to, poboljšanje rekonstrukcije moglo bi se postići točno definiranim kutovima akvizicije slike, postavljajući kameru na čvrstu konstrukciju. Na ovaj način bi bilo moguće izračunati točne transformacijske matrice, što bi uvelike pospješilo rekonstrukciju. Nadalje, uz bolje ulazne podatke, algoritam je moguće dodatno automatizirati, jer bi bilo manje potrebe za ručnim pregledavanjem i odabiranjem. Konačno, neuronska mreža bi davala preciznije rezultate kada bi se trenirala sa više točaka i uz puno više ulaznih podataka.

Prikupljeni i označeni podatkovni skup, algoritam za rekonstrukciju i neuronska mreža predloženi u ovom radu pokazali su da detekcija grana za rezidbu na temelju RGB-D slika je smislen postupak koji ima potencijala za daljnja poboljšanja, što čini temelj za budući razvoj

automatizacije rezidbe grana u voćnjacima.

LITERATURA

[1] Shayan A. Akbar, Somrita Chattopadhyay, Noha M. Elfiky, and Avinash Kak. A novel benchmark rgbd dataset for dormant apple trees and its application to automatic pruning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 347–354, 2016.

[2] Jing Zhang, Long He, Manoj Karkee, Qin Zhang, Xin Zhang, and Zongmei Gao. Branch detection for apple trees trained in fruiting wall architecture using depth features and regions-convolutional neural network (r-cnn). *Computers and Electronics in Agriculture*, 155:386–393, 2018.

[3] Manoj Karkee and B. Adhikari. A method for three-dimensional reconstruction of apple trees for automated pruning. *Transactions of the ASABE*, 58:565–574, 06 2015.

[4] Kong-man (German) Cheung, Simon Baker, and Takeo Kanade. Shape-from-silhouette across time part i: Theory and algorithms. *International Journal of Computer Vision*, 62(3):221–247, May 2005.

[5] Manoj Karkee, Bikram Adhikari, Suraj Amatya, and Qin Zhang. Identification of pruning branches in tall spindle apple trees for automated pruning. *Computers and Electronics in Agriculture*, 103:127–135, 2014.

[6] Qi Wang and Qin Zhang. Three-dimensional reconstruction of a dormant tree using rgb-d cameras. volume 2, 01 2013.

[7] Fu Yuxing, Li Chengming, Zhu Jiang, Wang Baolong, Zhang Bin, and Fu Wei. Three-dimensional model construction method and experiment of jujube tree point cloud using alpha-shape algorithm. *Transactions of the Chinese Society of Agricultural Engineering (Transactions of the CSAE)*, 36(22):214–221, 2020.

[8] Yaqoob Majeed, Jing Zhang, Xin Zhang, Longsheng Fu, Manoj Karkee, Qin Zhang, and Matthew D. Whiting. Apple tree trunk and branch segmentation for automatic trellis training using convolutional neural network based semantic segmentation. *IFAC-PapersOnLine*, 51(17):75–80, 2018. 6th IFAC Conference on Bio-Robotics BIOROBOTICS 2018.

[9] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

[10] Noha M. Elfiky, Shayan A. Akbar, Jianxin Sun, Johnny Park, and Avinash Kak. Automation of dormant pruning in specialty crop production: An adaptive framework for automatic reconstruction and modeling of apple trees. In *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 65–73, 2015.

[11] Robot operating system. https://ros.org/. Pristup: 3.2.2024.

[12] Open3d - a modern library for 3d data processing. https://www.open3d.org/. Pristup: 11.3.2024.

[13] Point cloud. https://www.open3d.org/docs/release/tutorial/geometry/pointcloud.html. Pristup: 12.3.2024.

[14] Global registration. https://www.open3d.org/docs/release/tutorial/pipelines/global_registration.html. Pristup: 12.3.2024.

[15] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017.

[16] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017.

[17] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[18] Open3d visualizer class. https://www.open3d.org/docs/release/python_api/open3d.visualization.Visualizer.html. Pristup: 3.5.2024.

[19] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J. Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1524–1531, 2014.

[20] Simulating kinect noise: adding noise to clean depth-maps rendered with a graphics engine. https://github.com/ankurhanda/simkinect. Pristup: 3.5.2024.

[21] 3d point cloud and mesh processing software. open source project. https://www.danielgm.net/cc/. Pristup: 3.5.2024.

SAžETAK

U ovom diplomskom radu predstavljen je prikupljeni podatkovni skup BRANCH koji sadrži RGB-D slike stabala voćki prije i poslije rezidbe. Predstavljen je i algoritam koji iz više dubinskih slika stabla rekonstruira 3D model stabla u obliku oblaka točaka. Također, opisana je i neuronska mreža temeljena na *PointNet++* arhitekturi. Algoritam za rekonstrukciju 3D modela podrazumijeva odabir oblaka točaka za rekonstrukciju, uklanjanje pozadinskog šuma i preklapanje oblaka točaka pomoću RANSAC i ICP algoritama. RANSAC se koristi za inicijalnu globalnu registraciju, a ICP za dodatno poravnavanje točaka. Evaluacija algoritma za rekonstrukciju 3D modela obavljena je na sintetičkom modelu. Korištene su metrike *Intersection over Union* - IoU i *chamfer distance*. Rezultat metrike IoU, za vrijednost praga od 1 cm, je 63.39%, a za vrijednost od 5 cm, rezultat je 94.66%. Metrika *chamfer distance* predstavlja prosječnu udaljenost između sparenih točaka skupa i ona iznosi 2.897 cm. Uspješnost izvedbe algoritma rekonstrukcije modela na BRANCH podatkovnom skupu iznosi 74.29%. Preklapanjem rekonstruiranih oblaka točaka drveća prije i nakon rezidbe dobivene su oznake točaka koje pripadaju orezanim granama. Ove oznake, zajedno s oblacima točaka drveća prije rezidbe koriste se kao ulaz za učenje neuronske mreže. Neuronska mreža je temeljena na *PointNet++* arhitekturi, na specifičnom modelu za segmentaciju dijelova objekta. Model mreže, program za učitavanje podatkovnog skupa i proces treniranja mreže su prilagođeni za potrebe diplomskog rada. Mreža je trenirana na skupu za učenje, a testirana na podatkovnom skupu za testiranje na kojem postiže točnost od 90.08%. Također na cijelom testnom skupu postiže preciznost 69.76%, odziv 67.85% i F1 68.79%. Upotreba kvalitetnijih senzora za prikupljanje slika sa što manje gubitka informacija te optimizacija algoritama za rekonstrukciju i klasifikaciju prijedlozi su za poboljšanje predložene metode za detekciju grana za rezidbu.

Ključne riječi:: detekcija grana, neuronska mreža, oblaci točaka, Python, RGB-D slike

ABSTRACT

Title: Detection of branches for pruning in RGB-D images.

In this thesis, the collected dataset BRANCH is presented, which contains RGB-D images of fruit trees before and after pruning. Additionally, an algorithm is introduced that reconstructs a 3D model of a tree from multiple depth images in the form of a point cloud. The thesis also describes a neural network based on the PointNet++ architecture. The algorithm for 3D model reconstruction involves selecting point clouds for reconstruction, removing background noise, and aligning point clouds using the RANSAC and ICP algorithms. RANSAC is employed for initial global registration, while ICP is used for further alignment of points. The evaluation of the reconstruction algorithm was performed on a synthetic model. The metrics used were Intersection over Union (IoU) and chamfer distance. The IoU result, with a threshold value of 1 cm, is 63.39%, and for a threshold of 5 cm, the result is 94.66%. The chamfer distance represents the average distance between paired points in the set, which is 2.897 cm. The algorithm's performance on the BRANCH dataset achieved 74.29%. By overlapping the reconstructed point clouds of trees before and after pruning, point labels corresponding to pruned branches were obtained. These labels, along with the point clouds of trees before pruning, are used as input for training the neural network. The neural network is based on the PointNet++ architecture, specifically tailored for object part segmentation. The network model, dataset loading program, and training process were adapted for the purposes of this thesis. The network was trained on the training set and tested on the test set, where it achieved an accuracy of 90.08%. Additionally, on the entire test set, it achieved a precision of 69.76%, recall of 67.85%, and an F1 score of 68.79%. The use of higher-quality sensors for image collection with minimal information loss, as well as the optimization of reconstruction and classification algorithms, are suggested for improving the proposed method for branch pruning detection.

Keywords: branch detection, neural network, point clouds, Python, RGB-D images

ŽIVOTOPIS

Andrej Bošnjak rođen je 5.4.1999. u Osijeku. Završio je II. gimnaziju Osijek. 2019. godine upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, smjer Računarstvo. Godine 2022. stječe akademski naziv sveučilišni prvostupnik (baccalaureus) inženjer računarstva. Iste godine upisuje diplomski sveučilišni studij računarstvo, smjer robotika i umjetna inteligencija. Tokom studija odrađuje dvije stručne prakse iz područja robotike i robotskog vida na zavodu za računalno inženjerstvo i automatiku. Koautor je rada *BRANCH - a Labeled Dataset of RGB-D Images and 3D Models for Autonomous Tree Pruning* koji će biti prezentiran na znanstvenom skupu International Conference on Smart Systems and Technologies 2024.

PRILOG

Github repozitorij koji sadrži programski kod AndrejBosnjak/DiplomskiRad_prilog.