Prepoznavanje voća na RGB-D slikama na temelju boje i geometrijskih značajki

Radočaj, Kristijan

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij

PREPOZNAVANJE VOĆA NA RGB-D SLIKAMA NA TEMELJU BOJE I GEOMETRIJSKIH ZNAČAJKI

Diplomski rad

Kristijan Radočaj

Osijek, 2017.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za obranu diplomskog rada

Osijek, 22.09.2017.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za obranu diplomskog rada

Ime i prezime studenta:	Kristijan Radočaj
Studij, smjer:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D 700 R, 14.10.2014.
OIB studenta:	92108696596
Mentor:	Prof.dr.sc. Robert Cupec
Sumentor:	Dr.sc. Ivan Vidović
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	Doc.dr.sc. Emmanuel-Karlo Nyarko
Član Povjerenstva:	Dr.sc. Ivan Vidović
Naslov diplomskog rada:	Prepoznavanje voća na RGB-D slikama na temelju boje i geometrijskih značajki
Znanstvena grana rada:	**Umjetna inteligencija (zn. polje računarstvo)**
Zadatak diplomskog rada:	Izraditi program za prepoznavanje voća na RGB-D slikama na temelju boje, koji kao predobradbu koristi segmentaciju dubinske slike na približno konveksne objekte.
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	22.09.2017.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 30.09.2017.

Ime i prezime studenta:	Kristijan Radočaj
Studij:	Diplomski sveučilišni studij Računarstvo, smjer Procesno računarstvo
Mat. br. studenta, godina upisa:	D 700 R, 14.10.2014.
Ephorus podudaranje [%]:	1 %

Ovom izjavom izjavljujem da je rad pod nazivom**: Prepoznavanje voća na RGB-D slikama na temelju boje i geometrijskih značajki**

izrađen pod vodstvom mentora Prof.dr.sc. Robert Cupec

i sumentora Dr.sc. Ivan Vidović

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj

1. UVOD

Branje voća predstavlja problem u automatizaciji i robotici jer su okolnosti u svakom trenutku jedinstvene i neponovljive u smislu da se svaki plod razlikuje od prethodnog ne samo sa svojim karakteristikama, poput veličine oblika, boje, već i karakteristikama svoje okoline. Trenutno se automatizacija koristi kod poslova koji zahtijevaju monotone ponavljajuće zadatke u kontroliranim uvjetima, to jest okolinama koje se ne mijenjaju. Robotima koji rade u stalnim okolinama moguće je programirati kretnje po unaprijed zadanoj putanji, no roboti koji rade u okolinama koje se mijenjaju svoje kretnje moraju samostalno računati u stvarnom vremenu. Dolaskom novih jeftinijih i pristupačnijih tehnologija, započelo je razvijanje automatizacije u područjima gdje su poslovi sklop jedinstvenih radnji s varijabilnom okolinom gdje se roboti moraju konstantno prilagođavati novim uvjetima i potrebama. Kako bi robot mogao pravilno i u cjelovitosti opažati okolinu u kojoj se nalazi potrebni su senzori. Jednostavnim senzorima, poput senzora udaljenosti, pokreta, svjetlosti itd., nije moguće u potpunosti i sa sigurnošću percipirati okolinu jer daju jednostavne i nepotpune podatke. Potrebe robota u varijabilnim okolinama zahtijevaju senzore koji mogu precizno, ali i cjelokupno opisati okolinu. 3D kamere mogu opisati okolinu na zadovoljavajućoj razini potrebnoj za lokalizaciju i orijentaciju robota u okolini. Dolaskom na tržište jeftinih, ali preciznih 3D kamera poput Microsoft Kinecta, Orbbec Perssea, ZED stereo kamere i PrimeSensa stvara se mogućnost njihovog korištenja kao senzora. Ovi senzori upravo zbog svoje pristupačnosti širokoj publici imaju i izgrađene podrške otvorenog koda koje omogućavaju bolji i brži razvoj korisničkih programa. Programska podrška poput *Point Cloud Library*-a(u daljnjem tekstu PCL) koja podržava nekoliko vrsta 3D kamera, te je građena da bude neovisna o platformi, odnosno operativnom sučelju, korisnicima olakšava kreiranje programa za raznovrsne primjene i područja.

Zadatak ovog diplomskog rada jest izrada programa za prepoznavanje plodova, u prirodnom okruženju, sa ciljem automatizacije branja voća. Kreiranjem baze podataka u koju se spremaju prijašnji rezultati prepoznavanja, te korištenjem istih u svrhu prepoznavanja na novim video snimkama ovom programu će se omogućiti učenje. Za izradu funkcija programa prepoznavanja korišten je PCL, a za predobradu videosnimki *Robot Vision Library* (u daljnjem tekstu RVL) koji je razvijen na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija (akronim FERIT). Također za klasifikaciju koristi se *Fast Library for Approximate Neasrest Neighbor*(u daljnjem tekstu FLANN). Izrađeni program potrebno je potom i testirati na setu videosnimki voća.

Drugo poglavlje ovog rada opisuje potrebnu predobradu sirovih ulaznih podataka, to jest pred obradu dubinskih slika za potrebe registracije objekata. U trećem poglavlju objašnjeni su

deskriptori, te dva pristupa opisivanju objekata pomoću kojih se vrši prepoznavanje. Četvrto poglavlje opisuje prepoznavanje objekata na sceni i algoritam kojim se to postiže. Peto poglavlje opisuje programsku implementaciju registriranja i prepoznavanja objekata, preciznije voća, na sceni. Šesto poglavlje prikazuje rezultate dobivene testiranjem programa, te usporedbu učinkovitosti dvaju pristupa opisivanja objekata.

2. SEGMENTACIJA NA PRIBLIŽNO KONVEKSNE POVRŠINE

Korištenje segmentacije na 3D slikama ima nekoliko prednosti spram segmentacije na 2D slikama, poput mogućnosti određivanja pozicije i orijentacije objekta spram senzora, veće robusnosti na zaklonjenost objekta, te informacije o dimenzijama objekta koje pomažu prilikom operacija kreiranja putanje bez kolizija i hvatanja objekta. Stoga poželjno primijeniti dubinske slike(engl. *range image*) nastale iz 3D senzora. Slika nastala iz ovakvih senzora osim vrijednosti za boju i intenzitet sadrži i podatak o udaljenosti piksela od neke točke najčešće senzora, te se stoga može nazvati 2.5D modelom scene.

Postupkom segmentacije vrši se razlaganje slike na segmente koji bi trebali predstavljati različite objekte. Postoji nekoliko metoda segmentacija koje se mogu podijeliti na dvije vrste. Metode na temelju rubova (engl. *Edge based*) i metode na temelju područja (engl. *Region based*). Segmentacije iz prve grupe uglavnom dijele sliku na segmente traženjem mjesta na kojima se boja, dubina, ili svjetlina naglo mijenjaju. Rubove možemo podijeliti na tri vrste. Prema [1 str. 610] prva vrsta ruba nastaje kada postoji procijep između dvije površine, druga nastaje na oštrom bridu objekta, a treća kada je rub glatki prijelaz između dvije ravnine. Ovi kriteriji podijele su dovoljni kada su prijelazi između objekata na slici vrlo izraženi, u prve dvije vrste rubova. Na segmentima gdje su prijelazi blagi, treća vrsta ruba, ova metoda ima tendenciju spajanja objekata.

Segmentacije iz druge skupine stvaraju segment na način da se točka koja je ili slučajno odabrana ili odabrana nekom od heurističkih metoda usporedi sa svim okolnim pikselima, te ako je zadovoljen uvjet sličnosti ti se pikseli slažu u grupe. Postupak se potom nastavlja za svaki rubni piksel grupe sve dok se ne pronađe rub objekta, to jest dok se niti jedan novi piksel ne može dodati u grupu. Ove metode na 2.5D slikama mogu segmentirati korištenjem samo ravnih ili sa kombinacijom ravnih i zaobljenih ploha. Grubo segmentiranje na ravne konveksne i konkavne površine izbjegava pogreške nastale zbog šuma.

Autori iz [2] koriste metodu za segmentaciju dubinskih slika na približno konveksne površine, gdje se prvo kreira 2.5D mreža trokuta(engl. *triangular mesh*) iz slike pomoću iterativne Deulanyeve triangulacije(engl. *iterative Deulany triangulation*). Potom se korištenjem inkrementalnog algoritma konveksne ljuske(engl. *incremental convex hull algoritm*) u kombinaciji sa širenjem područja(engl. *region growing*) dobivaju objekti podijeljeni na približno konveksne površine.

2.5D mreža je mreža trokuta kojom se aproksimiraju objekti sa poliedarskim površinama uz određenu pogrešku. Ako se mreža trokuta zapiše kao M u obliku skupa sa članovima F_k, koji predstavljaju trokute, tada ju možemo zapisati kao[2 str. 2]: $M=\{F_1, F_2, ..., F_\mu\}$. Mreža za koju

vrijedi da za svaki par trokuta F_i, $F_j \in M$ postoji niz $(F_{s_1}, F_{s_2}, \dots, F_{s_r})$, gdje je s_1=i, s_r=j i svaka dva uzastopna trokuta u nizu imaju zajedničku stanicu naziva se povezanom mrežom.

Svi vrhovi trokuta u mreži M se mogu označiti kao $\sum_M$. Ako za par točaka P, $P' \in \sum_M$ vrijedi[2 str. 3]:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \cdot (p' - p) \neq 0, \tag{2-1}$$

gdje $p=[x,y,z]^T$ označava vektor pozicije točke P, tada se mreža sa ovim svojstvima naziva još i 2.5D mrežom. Konveksna ljuska 2.5D mreže je također poliedarska površina i može se predstaviti skupom trokuta kao: $H_M=\{T_1, T_2, \dots, T_m\}$. Svakom se trokutu T_k dodjeljuje normala $\boldsymbol{n_k}$ usmjerena iz ljuske, te parametar ρ_k koji predstavlja udaljenost ravnine u kojoj leži trokut od ishodišta koordinatnog sustava. Ravnina na kojoj trokut T_k leži može se definirati jednadžbom[2 str. 3]:

$$\forall P \in T_k, \boldsymbol{n}_k^T \cdot \boldsymbol{p} - \rho_k = 0 \tag{2-2}$$

Ako se podskup od H_m^+ definira kao:

$$H_M^+ = \{T_k \in H_M : \boldsymbol{n}_k^T \cdot \boldsymbol{z} > 0\}, \tag{2-3}$$

a M dobiva iz 3D senzora, tada H_M^+ predstavlja podskup trokuta iz skupa H_M koji su vidljivi senzoru. Ako za povezanu mrežu M vrijedi:

$$\max_{P \in \sum_M} \left\{ \min_{T_k \in H_M^+} \{\xi(\rho_k - \boldsymbol{n}_k^T \boldsymbol{p})\} \right\} \leq \varepsilon \tag{2-4}$$

gdje je ξ =1 i ε je korisnički zadana tolerancija, tada se ta 2.5D povezana mreža naziva i približno konveksna površina sa odstupanjem ε.

Algoritam segmentacije trokutaste mreže može se opisati slijedećim koracima, prema [2 str. 59]. Prvi korak je odabir najvećeg trokuta. Ovaj će trokut predstavljati korijen, početni skup, postupka iterativnog širenja područja. Ovaj postupak radi na principu dodavanja susjednih trokuta, koji zadovoljavaju uvjet iz(2-4), trokutima iz skupa. Postupak dodavanja se ponavlja sve dok ima susjednih trokuta koji dijele stranicu sa barem jednim trokutom iz skupa i zadovoljavaju zadani uvjet. Algoritam se ponavlja sve dok na sceni postoje trokuti koji ne pripadaju nekom segmentu, te na kraju za rezultat daje skup $\sum$ segmenata približno konveksnih površina.

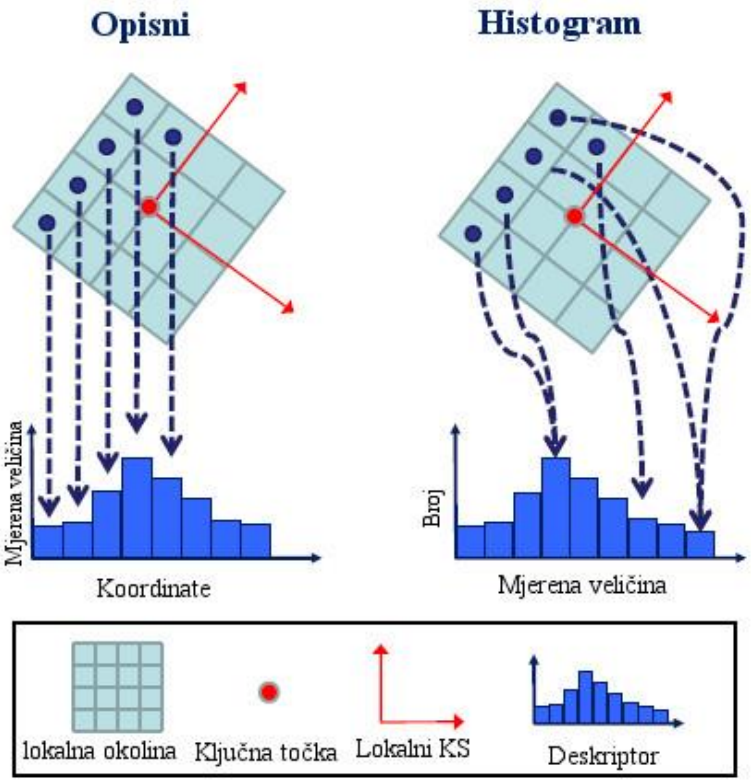
3. DESKRIPTORI KONVEKSNIH POVRŠINA

Deskriptor je struktura koja jedinstveno opisuje set podataka. U području računalnog vida oni opisuju vizualne odlike sadržaja na slici ili videu, poput oblika, boje, teksture. Osnovna podjela deskriptora odnosi se na pristup obradi ulaznih podataka i dijeli se na globalni i lokalni pristup. Globalni deskriptori opisuju geometriju objekta kao cjelinu i stoga je potrebna pred obrada, segmentacija, kako bi se scena podijelila na objekte. Neki od deskriptora koji koriste globalni pristup su: Matrice oblika(engl. *Shape Matrices*), Histogrami orijentiranih gradijenata(engl. *Histograms of Oriented Gradients*), Histogram značajki iz gledišta(engl. *Viewpoint Feature Histogram*) i Asambl oblika funkcija(engl. *Ensemble of Shape Functions*). Za razliku od globalnih deskriptora, lokalni opisuju neposrednu geometriju okoline individualne točke koje se zadaju kao ulaz, to jest ključne točke(engl. *keypoints*), i neovisni su o objektima na sceni. Uobičajeno je da korisnik sam bira ključne točke, stoga je veoma bitno da se odabiru one točke koje imaju jedinstvenu neposrednu okolinu. Neki od lokalnih deskriptora su: Histogram značajki točke (engl. Point Feature Histogram), Deskriptor površine na temelju radijusa(engl. *Radius-Based Surface Deskriptor*), 3D kontekst oblika (engl, *3D Shape Context*), Statistike rotacijske projekcije (engl. Rotational Projection Statistics) i Potpisi histograma orijentacija (engl. *Signatures of Histograms of Orientations*). Potrebno je napomenuti da se neki lokalni deskriptori mogu koristiti i kao globalni. To je moguće učiniti sa onim deskriptorima koji traže susjede pomoću radijusa. Potrebno je samo koristiti jednu ključnu točku po objektu te namještati radijus tako da obuhvati cijeli objekt.

3.1. Podjela deskriptora

Većinu lokalnih deskriptora koji su do danas izmišljeni moguće je svrstati u jednu od dvije osnovne kategorije: opisne (engl. *Signatures*) i histograme. Deskriptori iz prve skupine 3D okolinu ključne točke(u daljnjem tekstu: lokalna okolina) opisuju na način da definiraju lokalni referentni koordinatni sustav uz pomoć koordinata gotovo svake točke lokalne okoline. Iako ovaj način opisivanja može biti veoma precizan, male pogreške u definiranju lokalnog referentnog koordinatnog sustava ili male perturbacije u kodiranim osobinama mogu značajno izobličiti završni deskriptor. Histogrami opisuju okolinu prebrajanjem topoloških entiteta, poput vrhova ili površina trokuta mreže(Engl. *mesh triangle areas*), te slaganjem u histogram prema određenoj domeni, kao na primjer koordinata, zakrivljenosti ili kutova normala trokuta mreže. Ako su domena deskriptora koordinate, tada se i ove metode baziraju na lokalnom referentnom koordinatnom sustavu. Ali u drugim slučajevima, kao na primjer kada je domena razlika kutova između normala, potrebna je samo ponovljiva referentna os(Engl. *repeatable reference axis*). Ako

usporedimo ove dvije skupine, može se zaključiti da histogrami daju veći značaj robusnosti u odnosu na preciznosti opisivanja okoline ključne točke.



Slika 3.1: Usporedba opisnih i histogramskih deskriptora[3, str. 4]

3.2. Histogram distribucije normala

Kao što je prije navedeno, histogrami pridodaju veću pozornost robusnosti, te su stoga pogodniji za korištenje kod scena gdje postoje različite smetnje poput šuma, zamućenosti, djelomične prikrivenosti objekta. Uvjete u voćnjacima, za snimanje stereo kamerama, nije moguće urediti na način da se osigura odgovarajuće osvjetljenje ili da se izbjegne zaklanjanje plodova listovima, granama ili drugim plodovima. Nadalje, stereo kamere su prvenstveno namijenjene za rad u zatvorenom prostoru bez jakog izvora svjetlosti, kao što je dnevno svjetlo Sunca, pri konstantnom osvjetljenju. Stoga je prednost robusnosti pred preciznošću opisivanja okoline dobrodošla. Prednost histogramskih deskriptora jest i njihova matematička jednostavnost, što pak ubrzava obradu te omogućuje rad u približno stvarnom vremenu. Histogram distribucije normala u ovom radu distribuira vrijednosti z-komponente normala trokuta približno konveksnih površina. Broj binova je radi dodatne robusnosti korisnički moguće definirati, izvorno histogram se dijeli na 8 binova.

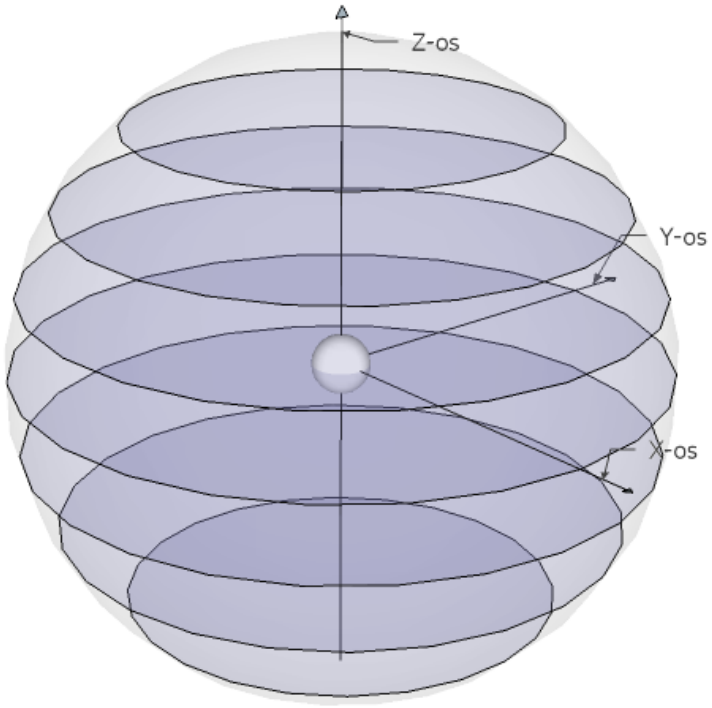
Algoritam kreiranja deskriptora moguće je podijeliti na dva glavna dijela. Prvi dio se odnosi na organiziranje ulaznih podataka i spremanje u matrice podataka. Drugi dio se odnosi na kreiranje histograma. Ulazni podaci su normale trokuta 2.5D mreže i brojevi točaka oblaka iz scene koji im pripadaju. Svaki trokut je predstavljen jednom normalom koja aproksimira normale svih njegovih točaka. Potom se računa ukupan broj točaka pojedinog objekta radi normalizacije što daje robusnost na rotaciju i translaciju objekata na sceni. Zatim slijedi kreiranje histograma. Vrijednosti z osi normala trokuta mreže mogu biti između minus jedan i jedan. S obzirom na to i na broj binova, odjeljaka, u histogramu svakom binu pripada određeni raspon, koji ovisi o broju binova, između te dvije vrijednosti. Algoritam prolazi kroz sve trokute segmenta te se ovisno o vrijednosti z-komponente normale trenutnog trokuta broj točaka koji pripada tom trokutu dodaje u bin pod čiji raspon vrijednost osi pripada. Potom se ukupan zbroj točaka u svakom binu dijeli sa ukupnim brojem točaka objekta čiji su trokuti u histogramu. Potom se postupak ponavlja za sljedeći segment. Neka je trokut konveksne površine T_k i ima normalu $\boldsymbol{n_k}$, te mu pripada **m** broj točaka oblaka. Ako histogram *H* sadrži korisnički određen *i* broj binova *b* tada svakom binu pripada raspon vrijednosti z-komponente:

$$b_j = [\frac{2j}{i} - 1, \frac{2(j+1)}{i} - 1 >, j = 0, \dots, i-2 \tag{3-1}$$

osim posljednjeg kojem, kako bi se obuhvatio cijeli raspon vrijednosti z-komponente, pripada raspon:

$$b_j = \left[1 - \frac{2}{i}, 1\right], j = i - 1 \tag{3-2}$$

Slika 3.2 grafički prikazuje podjelu orijentacije normale po z-osi točke oblaka. Na slici je jedinična kružnica podijeljena na 8 jednakih dijelova, po z-osi, i svaki odjeljak predstavlja jedan bin histograma.



Slika 3.2: Grafički prikaz histograma distribucije normala

3.2.1 Primjena histograma normala na RGB-D podatcima

Kada je riječ o histogramskim deskriptorima moguće je jednostavno dodati podatke o teksturi, u ovom radu taj podatak je boja lokalne okoline. Kako bi deskriptor ostao što jednostavniji, radi brzine izračuna, podatci o lokalnoj teksturi se svode na srednju vrijednost boja točaka oblaka lokalne okoline. Oblak točaka je u RGB prostoru boja, no u tom prostoru boja nije moguće odvojiti podatke o boji i podatke o svjetlini koje nisu pogodne za usporedbu u uvjetima gdje osvjetljenje nije stalno. Stoga je potrebno pretvoriti podatke o srednjim vrijednostima R, G i B kanala lokalne okoline u pogodniji prostor boja. YCbCr prostor boja se sastoji od tri kanala, osvjetljenja *Y*(engl. *iluminance*), razlike vrijednosti plave boje i osvjetljenja C_B, te razlike vrijednosti crvene boje i osvjetljenja C_R. Pretvaranje se vrši prema[4 str. 52]:

$$\begin{bmatrix} Y \\ C_B \\ C_R \end{bmatrix} = \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \frac{1}{256} \begin{bmatrix} 65{,}738 & 129{,}057 & 25{,}064 \\ -37{,}945 & -74{,}494 & 112{,}439 \\ 112{,}439 & -94{,}154 & -18{,}285 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3-3}$$

Prije pretvaranja se računaju srednje vrijednosti *R*, *G* i *B* kanala lokalne okoline, a potom se vrši transformacija. Kako se u YCbCr prostoru boja većina podataka o obliku nalazi u *Y* kanalu taj se kanal zanemaruje, te se računaju samo preostala dva. Dodatno je moguće pojednostaviti spremanje ovih podataka ako se dobivene vrijednosti kanala pretvore iz dekadskog sustava u heksadekadski

i potom se kodira u jedan broj(3.4), gdje lijeve, značajnije, dvije znamenke predstavljaju heksadekadsku vrijednost C_B, a desne dvije C_R kanala.

$$C_B C_{R_{hex}} = C_{B_{hex}} * 100_{hex} + C_{R_{hex}} \tag{3-4}$$

Ova se vrijednost potom dodaje histogramu kao nova vrijednost za posljednji bin.

3.3. SHOT

Algoritam SHOT deskriptora osmišljen je u prvom redu za primjenu u rješavanju potreba poklapanja površina kada je prisutno mnogo objekata na sceni i kada je traženi objekt djelomično zaklonjen. Estimacija normale na površinu izvodi se metodom najmanjih kvadrata(engl. *Total Least Squares*, TLS), koja se svodi na dekompoziciju kovarijantne matrice na svojstvene vrijednosti(engl. *EigenValue Decomposition*, EVD). Kovarijantna matrica $\boldsymbol{M}$ se računa na temelju k najbližih susjeda p_i točkaste značajke:

$$\boldsymbol{M} = \frac{1}{k}\sum_{i=0}^{k}(\boldsymbol{p}_i - \widehat{\boldsymbol{p}})\,(\boldsymbol{p}_i - \widehat{\boldsymbol{p}})^T,\ \ \widehat{\boldsymbol{p}} = \frac{1}{k}\sum_{i=0}^{k}\boldsymbol{p}_i \tag{3-5}$$

TLS estimacija smjera normale se računa pomoću svojstvenog vektora koji odgovara najmanjoj svojstvenoj vrijednosti matrice $\boldsymbol{M}$. SHOT ovaj algoritam izmjenjuje tako da onim točkama koje su udaljenije od značajke pridodaje manja težina, kako bi se povećala ponovljivost u prisustvu većeg broja objekata. Potom se sve točke unutar sferične okoline, radijusa R, koje se koriste za izračun deskriptora koriste i za izračun matrice $\boldsymbol{M}$. Ovaj korak pomaže u robusnosti na šum. Kako bi se postigla veća efikasnost zanemaruje se izračun središta i zamjenjuje se vrijednošću ključne točke $\boldsymbol{p}$.

$$\boldsymbol{M} = \frac{1}{\sum_{i:d_i \le R}(R - d_i)}\sum_{i:d_i \le R}(R - d_i)\,(\boldsymbol{p}_i - \boldsymbol{p})(\boldsymbol{p}_i - \boldsymbol{p})^T \tag{3-6}$$

Gdje je $d_i = \|\boldsymbol{p}_i - \boldsymbol{p}\|_2$. Prema [3 str. 7] svojstveni vektori matrice $\boldsymbol{M}$, imaju svojstvo ponovljivosti i u scenama s nagomilanim objektima te uz prisutnost mjernog šuma. Svojstveni vektori iz jednadžbe (3-6) omogućuju ponovljive smjerove za osi lokalnog referentnog koordinatnog sustava, no javlja se problem dvoznačnosti, koji se mora riješiti kako bi se dobio jedinstveni lokalni referentni koordinatni sustav. U ovom algoritmu se to postiže na način da se svaki svojstveni vektor, preorijentira kako bi mu smjer odgovarao većini vektora koje predstavlja. Znakovima $\mathbf{x}^+$, $\mathbf{y}^+$ i $\mathbf{z}^+$ se označuju tri svojstvena vektora, a sa $\mathbf{x}^-$, $\mathbf{y}^-$ i $\mathbf{z}^-$ su označeni njima suprotni vektori. Na primjer, ako je *M(k)* podskup točaka unutar lokalne okoline čija je udaljenost od točkaste značajke unutar k najbližih srednjoj udaljenosti d_m svih točaka okoline. Tada se predznak konačne osi $\mathbf{x}$ računa prema:

$$S_x^+ \doteq \{i: d_i \le R \wedge (\boldsymbol{p}_i - p) \cdot \boldsymbol{x}^+ \ge 0\} \tag{3-7}$$

$$S_x^- \doteq \{i: d_i \le R \wedge (\boldsymbol{p}_i - p) \cdot \boldsymbol{x}^- > 0\} \tag{3-8}$$

$$\tilde{S}_x^+ \doteq \{i: i \in M(k) \wedge (\boldsymbol{p}_i - p) \cdot \boldsymbol{x}^+ \ge 0\} \tag{3-9}$$

$$\tilde{S}_x^- \doteq \{i: i \in M(k) \wedge (\boldsymbol{p}_i - p) \cdot \boldsymbol{x}^- > 0\} \tag{3-10}$$

$$\boldsymbol{x} = \begin{cases} \boldsymbol{x}^+, & |S_x^+| > |S_x^-| \\ \boldsymbol{x}^-, & |S_x^+| < |S_x^-| \\ \boldsymbol{x}^+, & |S_x^+| = |S_x^-| \wedge \left|\tilde{S}_x^+\right| > \left|\tilde{S}_x^-\right| \\ \boldsymbol{x}^-, & |S_x^+| = |S_x^-| \wedge \left|\tilde{S}_x^+\right| < \left|\tilde{S}_x^-\right| \end{cases} \tag{3-11}$$

Gdje je:

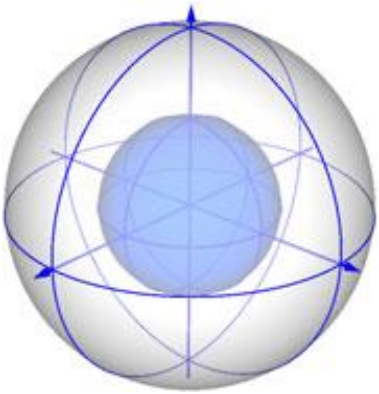
$$M(k) \doteq \{i: |m - i| \le k; m = \arg\operatorname*{median}_j d_j\} \tag{3-12}$$

Kako bi se razriješio predznak svojstvenog vektora kada su $|S_x^+| = |S_x^-|$ uzimaju se samo neparan broj vektora iz podskupa *M(k)* koji tvore nove podskupe $\tilde{S}_x^+$ i $\tilde{S}_k^-$, a konačni se vektor orijentira prema većini iz ta dva podskupa. Potom se ista procedura izvodi i za **z** os, a **y** os se računa kao **y**×**x**.

Ideje za SHOT zapravo dolaze iz 2D deskriptora zvanog transformacija značajki neosjetljiva na skaliranje(engl. *Scale-Invariant Feature Transform*, SIFT). Proučavajući SIFT moguće je vidjeti da se histogrami koriste u nekoliko koraka algoritma, od definicije lokalnih orijentacija pa sve do samog deskriptora. Također, kako jedan histogram na cijeloj lokalnoj okolini ne bi bio dovoljno deskriptivan, SIFT se oslanja na set lokalnih histograma koji se računaju na specifičnom podskupu piksela definiranih koordinatnom mrežom(engl. *grid*) položenom na lokalnu okolinu. Konačno, lokalni histogrami opisuju razdiobu derivacije prvog reda signala, poput gradijenata intenziteta.

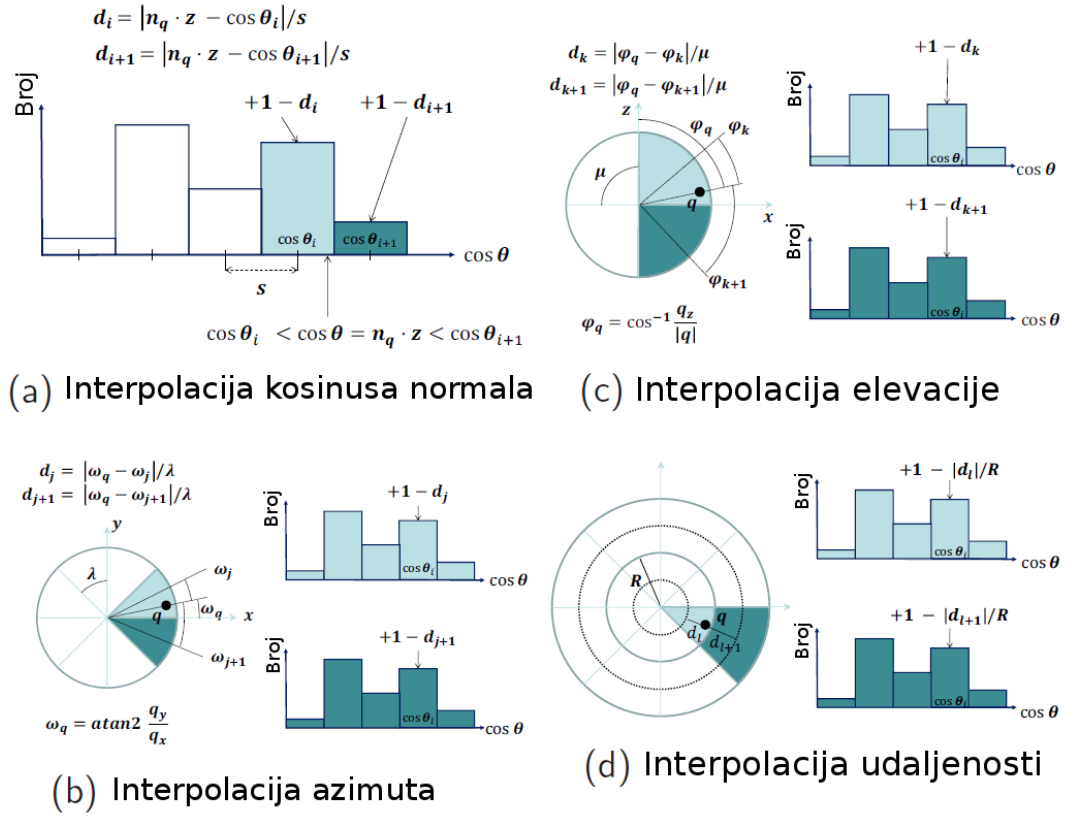
Prema uzoru na SIFT, SHOT koristi histograme normala voksela 3D lokalne okoline. Ovi se deskriptori zasnivaju na geometrijskim informacijama o položaju točaka unutar lokalne okoline, što oponaša potpis. Kod SHOT-a se to postiže računanjem skupa lokalnih histograma unutar 3D volumena definiranih 3D koordinatnom mrežom položenom na lokalnu okolinu. Mreža se poravnava prema osima lokalnog referentnog okvira, spomenutima ranije u tekstu.

Za svaki lokalni histogram akumuliraju se točke u pododjeljke ovisno prema kosinusu kuta θ_q između normale u točki $\boldsymbol{n_q}$ i osi **z** u točkastoj značajki $\mathbf{z}_k$. Razlozi korištenja kosinusa su mogućnost brzog računanja, jer vrijedi $cos\theta_q = \boldsymbol{z}_k \cdot \boldsymbol{n}_q$. I zbog toga što jednako razmaknute podijele $\cos\theta_q$ odgovaraju prostorno promjenjivim podjelama θ_q, što uzrokuje da male razlike u ortogonalnim smjerovima normala akumuliraju vrijednosti u različitim stupcima histograma.



Slika 3.3: Potpisna struktura SHOT deskriptora [5 str. 363]

Za potpisnu strukturu koristi se izotropna kuglasta rešetka(engl. *isotropic spherical grid*) koja obuhvaća podijele duž radijalne osi, te azimuta i elevacije, kako je prikazano slikom 3.3. Volumen svakog pododjeljka definira vrlo deskriptivan entitet predstavljen histogramom.



Slika 3.4: Četverostruka linearna interpolacija radi akumuliranja težina u histograme[4 str. 11]

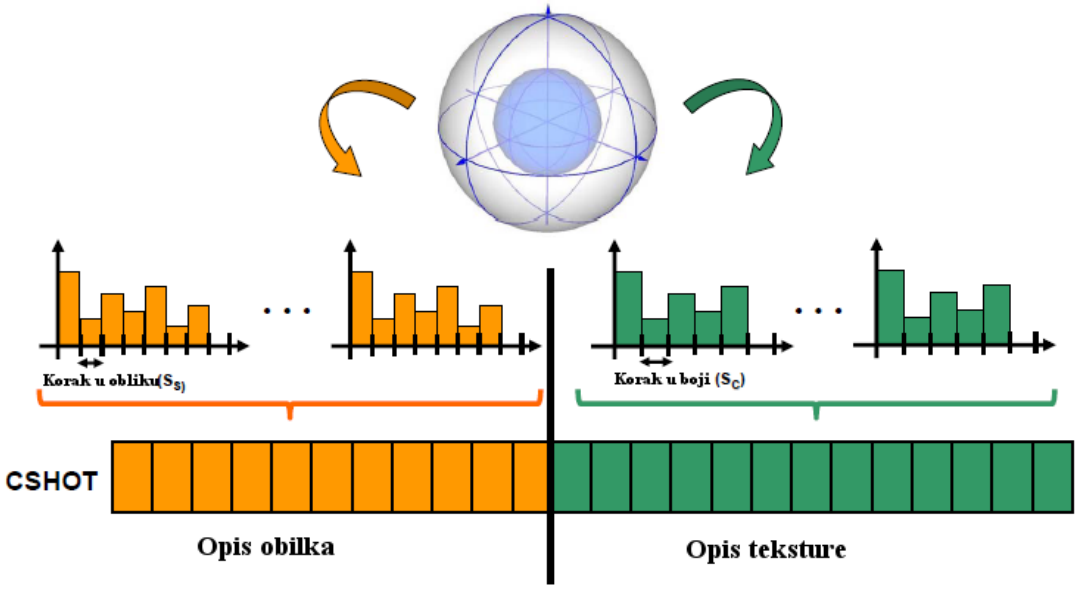
Odgovarajući broj volumena prema eksperimentu tvoraca je 32, što rezultira sa osam podjela po azimutu, dvije podjele po elevaciji i dvije podjele po radijusu. Pomnoži li se sa 11 internih histograma, ukupna dužina deskriptora je 352.

Kako bi se izbjegle pogreške zbog rubnih vrijednosti izvodi se četverostruka interpolacija sa susjednim stupcima, kako je prikazano slikom 3.4. Svaki stupac se uvećava za težinu u iznosu od 1-*d*, gdje je *d* udaljenost trenutnog unosa od središnje vrijednosti stupca histograma. Tako je za azimut i elevaciju *d* iznos kutne razlike trenutnog unosa i središnje vrijednosti volumena pododjeljka, a kod radijalne veličine *d* je euklidska udaljenost trenutnog unosa i središta pododjeljka. Mjerenje je normalizirano unutar svake dimenzije udaljenošću dvaju stupca histograma, to jest volumena pododjeljka.

Kako bi se postigla robusnost na promjenu gustoće točaka u oblaku točaka vrši se euklidska normalizacija deskriptora kako bi imao euklidsku normu jednaku jedan. To se radi na način da se svaki stupac histograma dijeli lokalnim brojem točaka.

3.3.1 Primjena SHOT-a na RGB-D oblak točaka

Kao što je prikazano slikom 3.5 vidljivo je da SHOT deskriptor može jednostavno uklopiti i informacije o teksturi lokalne okoline značajke, poput boje. U generalizaciji deskriptora dva histograma numerički opisuju oblik i teksturu deskriptora u danoj točki.

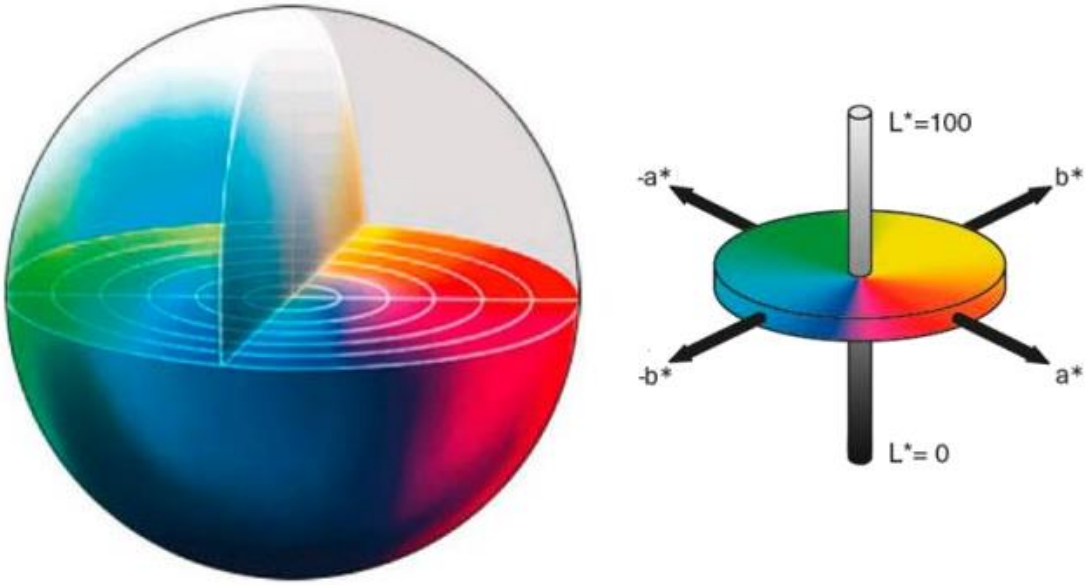


Slika 3.5: Izgled SHOT deskriptora za RGB-D slike[6 str 809.]

Kako bi se dobio deskriptor za teksturu potrebno je kreirati obilježja točaka, takva da se mogu usporediti s značajkom *k* i sa svakom točkom unutar lokalne okoline, te je potrebno kreirati određenu vrstu mjerne jedinice pomoću koje bi se mogle usporediti dvije takve vrijednosti.

SHOT se bazira na L_p normalizaciji između dviju trojki podataka, gdje svaka trojka predstavlja jednu boju. Točnije, implementiran je operator na bazi L_1 norme koji se sastoji od sume apsolutnih

vrijednosti razlika između trojki točkaste značajke i točke lokalne okoline. Također, umjesto korištenja RGB trojke koristi se CIELab prostor boja, slika 3.6. Ovaj prostor boja je temeljen na objektivnom vrednovanju boja i najbliži je vizualnoj percepciji[7]. Sastoji se od tri komponente svjetline *L*, zeleno-crvene *a* i žuto-plave *b* kromatske osi. Usporedba između dviju CIELab trojki podataka vrši se na temelju L_1 norme.



Slika 3.6 CIELab prostor boja[7]

4. KLASIFIKACIJA KONVEKSINH POVRŠINA

U ovom poglavlju govoriti će se o razvrstavanju segmenata na sceni u svrhu prepoznavanja objekata od interesa. Pojednostavljeno govoreći, prepoznavanje objekata na sceni vrši se usporedbom svakog segmenta sa već poznatim modelom objekta. U slučajevima kada se traži istovjetan objekt na sceni sa modelom ovo je jednostavna usporedba dvaju deskriptora sa korisnički predodređenim pragom tolerancije. Kada je riječ o prepoznavanju voća na sceni ovakav pristup nije zadovoljavajući zbog toga što se dva ploda iste voćke mogu veoma razlikovati. Stoga je potreban drugačiji pristup ovom problemu, pristup koji je dovoljno robustan na varijacije oblika i veličine. Pristup klasifikacije objekata nudi dovoljnu robusnost za zadovolji potrebe problema predstavljenog u ovom radu.

Klasifikacija je tehnika učenja koja pokušava naučiti izvršiti zadanu funkciju, na temelju skupa podataka za učenje. Ova funkcija se potom koristi za predviđanje klase na temelju ulaznih podataka. Skup podataka za učenje sastoji se od parova ulaznog i izlaznog podatka, gdje je ulazni podatak značajka objekta, a izlazni pripadajuća klasa. Neke od poznatijih metoda klasifikacije su logistička regresija(engl. *logistic regression*), Fisherova diskriminantna analiza i klasifikator k najbližih susjeda(engl. *k-nearest-neighbor classifier*). Klasifikator k najbližih susjeda je jedan od najjednostavnijih algoritama za učenje. Zanimljiv je jer računa funkciju lokano, to jest za svaki slučaj zasebno, te se računanje vrši pri klasifikaciji. Iako je jednostavan, njegovi rezultati su i u usporedbi sa novijim i kompleksnijim algoritmima vrlo dobri, prema izvoru[8]. Izvorni oblik algoritma namijenjen je jednodimenzionalnim ulaznim parametrima. Stoga se u ovom radu koristi izmijenjena inačica prilagođena za rad sa visoko dimenzijskim varijablama.

4.1. FLANN

FLANN, je biblioteka algoritama za brzo traženje približnih najbližih susjeda(engl. *fast approximate nearest neighbor search*) u visoko dimenzijskim prostorima.

Problem pretraživanja najbližih susjeda može se definirati na sljedeći način: ako postoji skup točaka $P=\{p_1, \ldots, p_n\}$ u vektorskom prostoru X i točka koja se želi klasificirati $q \in X$, potrebno je pred obraditi skup P kako bi se točke najbliže q mogle efikasno pronaći. FLANN ima ugrađenu automatizaciju izbora i konfiguracije algoritma na temelju skupa podataka za učenje. Koristi algoritam za nasumično pretraživanje k dimenzijskih stabala, koji održava isti redoslijed prioriteta pretraživanja grana preko svih stabala kako bi se pretraga organizirala na temelju udaljenosti između granica binova, pri čemu svaki bin predstavlja jednu dimenziju varijable. Također koristi

algoritam za pretraživanje zasnovan na grupiranju *k* srednjih vrijednosti(engl. *k-menas*) koji koristi redoslijed prioriteta pretraživanja domena za širenje pretrage na temelju udaljenosti *k* srednje vrijednosti domene od točke *q*.

Izvorni algoritam za stvaranje *k* dimenzijskog, kraće *kd*, stabla skupa za učenje je efikasan na skupovima niskog broja dimenzija ali njegova efikasnost naglo opada s porastom broja dimenzija vektorskog prostora *X*. Stoga FLANN koristi unaprijeđenu inačicu algoritma nazvanu nasumično kd-stablo(engl. *randomized kd-tree*). Ova inačica nasumično dijeli dimenzije iz prvih D dimenzija gdje varijabla ima najveću varijancu, a u FLANN-u je D statično postavljen na vrijednost od 5 dimenzija. Stupanj nasumičnosti se određuje promatranjem fiksnog broja krajnjih čvorova na kojima pretraživanje završava, te vraćanjem najboljih rezultata. Potrebno je samo korisnički odabrati željnu preciznost pretraživanja.

Hijerarhijsko stablo *k* srednjih vrijednosti nastaje podjelom varijable na *K* različitih područja grupiranjem *k* srednjih vrijednosti, te potom primjenom iste metode za sve binove u svakom području. Stvaranje stabla se zaustavlja kada je broj binova u regiji manji od *K*. Autori FLANN-a su stvorili algoritam koji pretražuje stablo na način da prvo prođe preko stabla i u red prioriteta dodaje sve neistražene grane u svakom čvoru. Potom uzima granu koja ima najbliži centar točki *q*, te se postupak ponavlja od te grane. Stupanj nasumičnosti je definiran jednako kao i u prijašnjem algoritmu.

Koji od ta dva algoritma će FLANN koristiti ovisi o nekoliko faktora poput strukture skupa podataka za učenje i tražene preciznosti pretraživanja. Za odluku boljeg algoritma koristi se cijena korištenja, koja se računa kao kombinacija vremena pretraživanja, vremena kreiranja stabla i korištenja memorije. U ovisnosti što je bitnije vrijeme ili memorija, cijena se može prilagoditi sa varijablama w_b - težina vremena stvaranja stabla, i w_m - težina korištenja memorije. Cijena *C* se računa prema[8 str. 4]:

$$C = \frac{s + w_b b}{(s + w_b b)_{opt}} + w_m m \tag{4-1}$$

Gdje je s vrijeme pretrage stabla, b je vrijeme izgradnje stabla i $m = m_t/m_d$predstavlja omjer memorije m_t korištene za stablo i memorije m_d korištene za spremanje podataka. Težina vremena stvaranja stabla ima raspon između nula i jedan, te predstavlja koliko je važno vrijeme stvaranja stabla u odnosu na vrijeme pretraživanja stabla. Ako se stablo stvara samo jednom, a koristi za sva pretraživanja tada se w_b može postaviti na nulu jer neće mnogo utjecati na ukupno vrijeme, no ako se stablo kreira za svaki slučaj tada je vrijeme izgradnje jednako bitno kao i vrijeme pretraživanja i potrebno je w_b postaviti na 1. Težina korištenja memorije također ima raspon između nula i jedan. Kada se postavi na nulu izbor algoritma neće ovisiti o korištenju memorije već samo o omjeru

brzine algoritma i optimalnoj brzini kreiranja stabla i pretraživanja(4-2), kada korištenje memorija nije faktor. Ako se pak postavi na jedan, tada potrošnja memorije i porast vremena kreiranja i pretrage stabla nose jednaku težinu pri izboru algoritma. Optimalna brzina kreiranja stabla i pretraživanja prema [8 str. 4]

$$(s + w_b b)_{opt} \tag{4-2}$$

Optimizacija parametara algoritma se može vršiti na cijelom skupu za učenje ili samo na jednom dijelu. Korištenjem cijelog skupa dobiju se najtočniji rezultati, no na velikim skupovima to uzima mnogo vremena. Autori FLANN-a su eksperimentiranjem došli do zaključka da se korištenjem desetine skupa postižu rezultati optimalno blizu rezultata dobivenih na cijelom skupu.

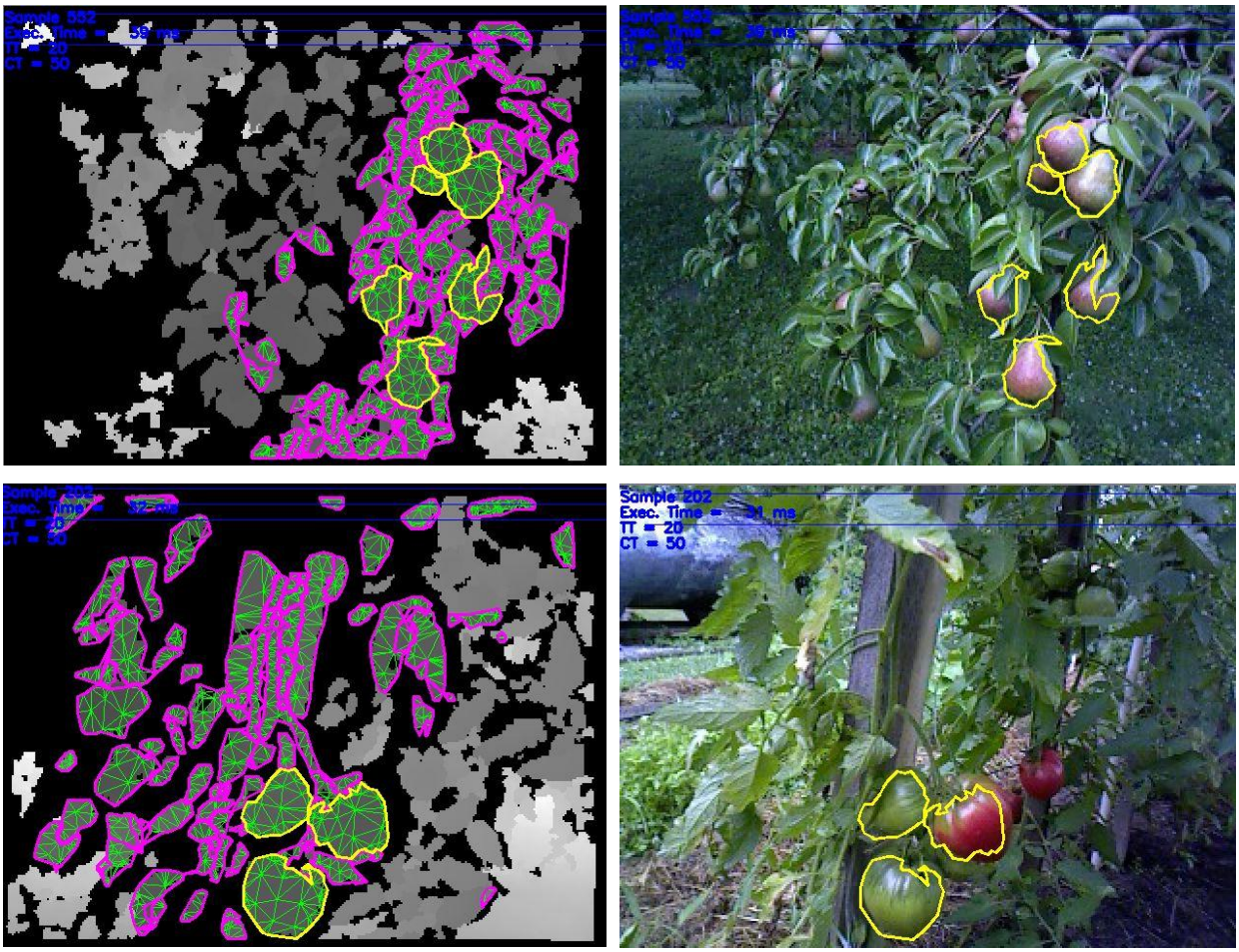
5. PROGRAM ZA DETEKCIJU VOĆA

Program za detekciju voća izrađen je u C++ programskom jeziku, sa sučeljem SQLite za rad s bazama podataka radi spremanja rezultata i stvaranja skupa za učenje. Koristi nekoliko biblioteka za rad. PCL se koristi za rad sa SHOT deskriptorom. dio PCL-a je i FLANN koji se koristi za klasifikaciju.

Program se može podijeliti na nekoliko dijelova. Prvi dio programa priprema segmentirane ulazne podatke za stvaranje deskriptora. U drugom dijelu se kreiraju dvije vrste deskriptora, histogram smjerova normala i SHOT za svaki segment. Potom slijedi dio u kojem se, ako je scena odabrana za učenje, svi deskriptori spremaju u bazu podataka, s pridruženom pripadnošću odgovarajućoj klasi. Klase koje se razmatraju u ovom radu su *voće* i *ostali objekti*. Scene, koje pripadaju skupu za učenje, su prethodno odabrane slučajnim odabirom. Sljedeći dio je klasifikacija objekata na scenama za testiranje i eksperimentalna evaluacija.

5.1. Segmentacija na približno konveksne površine

Skup podataka nad kojima se vrši obrada u ovom radu su video snimke snimljene PrimeSence Carmine 1.09 kamerom. Svaka scena snimki predstavljena je dubinskom slikom i RGB slikom. Dubinska slika ne sadrži informaciju o objektima, već samo o udaljenostima pojedine točke oblaka od kamere, stoga je potrebna predobrada. Dubinska slika se segmentira na približno konveksne površine metodom opisanom u [2, 10]. Prije segmentacije prvo se površine aproksimiraju kreiranjem 2.5D mreže trokuta. Na slici 5.1*(lijevo)* mreža trokuta prikazana je zelenim linijama.



Slika 5.1: Dubinske slike voća(*lijevo*) sa pred obradom segmentacije na približno konveksne trokute i (*desno*) njihovi RGB predstavnici

Segmentacija na približno konveksne površine vrši se na mreži trokuta i primjer je prikazan slikom 5.1(*lijevo*) gdje ljubičaste linije predstavljaju segmente.

5.2. Struktura programa

Funkcija programa razvijenog u okviru ovog rada je ugrađena unutar aplikacije za prikaz i segmentaciju dubinskih slika nazvane *RVLPCSdemo*. Ova aplikacija kao ulaz prima parove RGB slike i dubinske slike iz snimki snimljenih 3D kamerom pomoću funkcije *VS.m_Kinect.GetImages*. Potom nad učitanom dubinskom snimkom vrši segmentaciju na približno konveksne površine pomoću funkcije *VS.Segment*. Obije navedene funkcije pripadaju klasi CRVLPCSVS koja predstavlja alat za segmentaciju oblaka točaka unutar RVL biblioteke.

Nakon segmentacije se poziva funkcija *ComputeDescriptors* izrađena u okviru ovog rada. Ova funkcija za ulaz ima dubinsku sliku, RGB sliku i 3D oblak točaka. Ako ulazni podatci pripadaju sceni koja je dio skupa scena za učenje, tada funkcija kao rezultat daje skup deskriptora s

pripadajućim klasama koje se spremaju u bazu podataka. U slučaju kada scena pripada skupu za testiranje, tada je rezultat funkcije datoteka u kojoj su zapisani rezultati klasifikacije programa.
Aplikacija također omogućuje i interaktivnu vizualizaciju segmentacije, te korisnički odabir segmenata funkcijom *GUI.InteractiveVisualization* iz klase CRVLPCSGUI. Slike primjera segmentacije su proizašle kao rezultat ove funkcije, a korisnički odabir segmenata se očituje žutim linijama na slikama.

5.3. Klase podataka

Ulazni podaci koji se koriste u programu su normale trokuta površine, te točke iz 2.5D oblaka. Normale nisu spremljene u jednu varijablu već su spremane za svaki trokut zasebno sa ostalim parametrima trokuta. Kako bi ih se moglo koristiti potrebno ih je prvo spremiti u jednu strukturu podataka. Izgled klase normala je sljedeći:

```
class Normal
{
public:
        double x, y, z;
        int dots;
        Normal(int mD=0, double mX=0.0, double mY=0.0, double mZ=0.0) : dots(mD), x(mX), y(mY), z(mZ) {}
};
```

Klasa sadrži četiri varijable. Varijable *x*, *y* i *z*, predstavljaju koordinate vektora normale trokuta, a *dots* predstavlja broj točaka koje pripadaju trokutu.
Kako bi se mogle odrediti ključne točke za svaki segment potrebno je pred obraditi točke scene. Zbog toga što nije svakoj točki 2D slike pridružena dubina, ključne točke se traže na 2D slikama. One su spremljene u klasu oblika:

```
class PixelPoint
{
public:
        int u = 0.0, v = 0.0, Y = 0, Cb = 0, Cr = 0, CbCr=0;
        unsigned int r = 0, g = 0, b = 0;
        PixelPoint(int mU = 0.0, int mV = 0.0, unsigned int mR=0, unsigned int mG=0, unsigned int mB=0) :u(mU), v(mV), r(mR), g(mG), b(mB)
```

```
        {}
};
```

Varijable *u* i *v* predstavljaju koordinate tog istog piksela na 2D slici. Varijable *r*, *g* i *b* predstavljaju kanale boja u GRB prostoru boja, a *Y*, *Cb* i *Cr* kanale u YCbCr prostoru boja.

5.4. Priprema podataka

Priprema podataka se vrši u funkciji *VS.Segment* koja se nalazi u aplikaciji *RVLPCSdemo*. Ulazni podaci u program u svom izvornom formatu nisu primjereni za izradu deskriptora, stoga je potrebno formatirati ih u primjeren format za daljnju uporabu. Program se dijeli na dva deskriptora, te je iz tog razloga i priprema ulaznih podataka podijeljena na dva dijela. Jedan od ulaznih parametara u program su trokuti 2.5D mreže na sceni, a dio varijabli koje ti objekti sadrže potreban je za izračun histograma distribucije normala. Preciznije, potrebne su dvije informacije: srednja vrijednost orijentacija normala i broj pripadajućih točaka trokuta. Također, trokuti nisu organizirani prema njihovoj pripadnosti segmentu te ih je potrebno i poredati. Prvo je potrebno proći kroz sve trokute i spremiti podatke u matricu gdje broj retka odgovara broju segmenta kojem trokut pripada. Svaki redak sadrži koordinate smjera normale trokuta, te broj točaka oblaka čije smjerove ta normala predstavlja. Radi normalizacije histograma kao nazivnik koristi se ukupan broj točaka 3D oblaka točaka segmenta, stoga je potrebno zbrojiti ukupan broj točaka koji pripada određenom segmentu. Za računanje histograma distribucije normala na RGB-D scenama potrebna je dodatna obrada. Tako se u prvom koraku u matricu još sprema i ukupan zbroj vrijednosti svakog kanala u RGB prostoru boja za sve točke segmenta, a tijekom zbrajanja broja točaka segmenta vrši se i izračun srednjih vrijednosti kanala u RGB prostoru boja, te potom i transformacija u YCbCr prostor boja.

SHOT deskriptor za ulazne podatke traži ključne točke svakog segmenta i 3D oblak točaka scene. Ulazni podatci u program su već navedeni trokuti i 3D oblak točaka, što znači da za potrebe SHOT-a je potrebno samo izračunati ključne točke kao pripremu. Za ključnu točku segmenta odabrana je središnja točka segmenta iz 3D oblaka točaka. Kako bi se ona pronašla potrebno je proći kroz sve točke segmenta, no kako ulazni podatci nisu segmenti već trokuti, prvo je potrebno organizirati podatke tako da se omogući jednostavna obrada. Osim toga pronalazak srednje točke u 3D prostoru na konveksnim površinama je složen, no pronalazak srednje točke, piksela, u 2D prostoru se svodi na izračun srednjih vrijednosti dviju koordinata, te se stoga i izračun ključne točke vrši na RGB slici. Nakon pronalaska srednje točke na masci provjerava se ima li ona pripadajuću točku u 3D

oblaku točaka, ako ima ta se točka sprema kao ključna točka, no ako ona ne postoji potrebno je pronaći najbližu točku na masci koja ima pripadajuću točku u oblaku točaka. Rezultati pripreme podataka su matrica normala trokuta i brojeva točaka koje pripadaju tim trokutima, te oblak točaka koji sadrži ključne točke segmenta.

5.5. Izračunavanje deskriptora

Izračun histograma distribucije normala prikazan je sljedećim blokom:

```
mHist.resize(xSize);
for (int i = 0; i < xSize; i++)
{
	mHist[i].resize(buc);
	mHist[i].assign(buc, 0.0);
	for (int j = 0; j < ySize[i]; j++)
	{
		for (int k = 0; k < buc - 1; k++)
		{
			if (M[i][j].z < ((2 * (k + 1)) / ((double)buc) - 1) && M[i][j].z >=
((2 * (k)) / ((double)buc) - 1))
			{
				mHist[i][k] += (M[i][j].dots / maks[i]);
			}
		}
		if (M[i][j].z <= 1 && M[i][j].z >= (1 - (2 / ((double)buc))))
		{
				mHist[i][buc - 1] += (M[i][j].dots / maks[i]);
		}
	}
}
```

Struktura podataka *mHist* je vektor vektora histograma gdje svaki red predstavlja jedan histogram korisnički definiranog broja binova, varijabla *buc*. Prije izračuna histograma potrebno je odrediti broj segmenata na sceni, *xSize*, te alocirati dovoljan broj vektora, nije potrebno odmah alocirati memoriju jer vektori dozvoljavaju promijene u veličinama varijabli. Alokacija memorije za svaki histogram se vrši netom prije njegovog izračuna naredbom *resize*. Naredba *assign* služi za inicijalizaciju svih binova histograma na početnu vrijednost. Svakom segmentu *i*, iz matrice *M*, pripada *j* trokuta s normalama. Program za izračun histograma uzima vrijednost z-osi svake normale tih trokuta te provjerava kojem rasponu pripada prema formulama (3-1) i (3-2), i u ovisnosti o rezultatu dodaje vrijednost broja točaka trokuta, *dots*, podijeljenu s ukupnim brojem točaka segmenta *maks* u odgovarajući bin histograma. Postupak se ponavlja sve dok ima neobrađenih normala segmenta, nakon čega se prelazi na sljedeći segment.

Prvi korak izračuna deskriptora je računanje normala prikazano sljedećim blokom:

```
pcl::NormalEstimationOMP<PointType, NormalType> norm_est;
	norm_est.setNumberOfThreads(4);
	norm_est.setInputCloud(scena);
```

```
        norm_est.setSearchMethod(tree);
        norm_est.setRadiusSearch(1.0e+2f);
        norm_est.compute(*normals);
```

Klasa *NormalEstimationOMP* dio je PCL biblioteke i služi za izračun normala oblaka točaka s omogućenom paralelizacijom. Funkcija *setNumberOfThreads* postavlja željeni broj niti koji će se koristiti za izračun, potom *setInputCloud* učitava oblak točaka nad kojim će se izvršiti izračun. Funkcija setSearchMethod postavlja metodu kojom se pretražuju najbliži susjedi, koja je u ovom slučaju stablo. Funkcija setRadiusSearch postavlja vrijednost najveće udaljenosti susjedne točke od one za koju se računa normala i u ovom radu iznosi 10 milimetara. Posljednja funkcija bloka, compute, vrši izračun i za rezultat daje 3D oblak normala.

Nakon izračuna normala slijedi blok za izračun SHOT deskriptora:

```
pcl::SHOTEstimationOMP<PointType, NormalType, DescriptorType> descr_est;
        descr_est.setNumberOfThreads(4);
        descr_est.setInputCloud(keypoints);
        descr_est.setInputNormals(normals);
        descr_est.setSearchSurface(scena);
        descr_est.setSearchMethod(tree);
        descr_est.setRadiusSearch(3.0e+2f);
        descr_est.compute(*descriptors);
```

Klasa *SHOTEstimationOMP* služi za izračun SHOT deskriptora s omogućenom paralelizacijom, također je dio PCL biblioteke. Kao oblak točaka nad kojim se vrši izračun postavlja se oblak sa ključnim točkama. Funkcija *setInputNormals* učitava oblak normala, a setSearchSurface učitava oblak točaka scene nad kojim će se tražiti najbliži susjedi ključnih točaka. Najveća udaljenost susjeda od ključne točke koja utječe na izračun deskriptora iznosi 30 milimetara. Funkcija *compute* kao rezultat izračuna daje oblak SHOT deskriptora.

5.6. Trening i test metoda

Voće nije moguće klasificirati izravnom usporedbom modela i objekta na sceni. Postoje tri metode strojnog učenja iz perspektive točnosti klasifikacije[11 str. 115]: višestruka provjera(engl. *cross-validation*), *bootstrap* i treniranje i test. U ovom radu koristi se metoda treniranja i testiranja čiji je princip dati programu uzorak podataka, skup za treniranje, kako bi se postavilo pravilo klasifikacije, te potom promatranje programa u radu radi provjere stope pogrešaka koju uzrokuje na podacima nad kojima program nema predznanje o klasi.

Metoda kao što joj i ime kaže se sastoji od dva glavna dijela treniranja programa i testiranja pravila. Za potrebe treniranja zahtijeva se uzorak podataka. Taj uzorak u ovom radu je baza parova podataka. Svaki par predstavlja jedan segment i sadrži deskriptor segmenta, te odgovor, u Boolovoj logici, na pitanje je li taj segment voće, koji ručno odabire korisnik. Kako bi se izradio

skup podatak za učenje program mora proći kroz 2.5D slike koje su predodređene za uzorak, izračunati deskriptore za sve segmente na slikama, te ih spremiti u bazu zajedno s odgovarajućom klasom u skupu za treniranje. Kada je baza izrađena, program prelazi na drugi korak, to jest testiranje. U ovom dijelu program prolazi kroz drugi dio 2.5D slika, namijenjen za testiranje, nad kojima nema podataka o klasi segmenta, te uz pomoć FLANN-a klasificira svaki segment, potom se taj rezultat uspoređuje s rezultatom ručne klasifikacije, koja predstavlja činjenično stanje.

6. EKSPERIMENTALNA EVALUACIJA

Eksperimentalnom evaluacijom želi se odrediti točnost programa za prepoznavanje voća na RGB-D slikama. Evaluacija je provedena na skupu RGB-D slika krošnji različitih voćki snimljenih Microsoft Kinect kamerom. On sadrži 2974 2.5D slike krošnji s vidljivim voćem više vrsta voćki, šljiva, nektarina, kruški i rajčica.

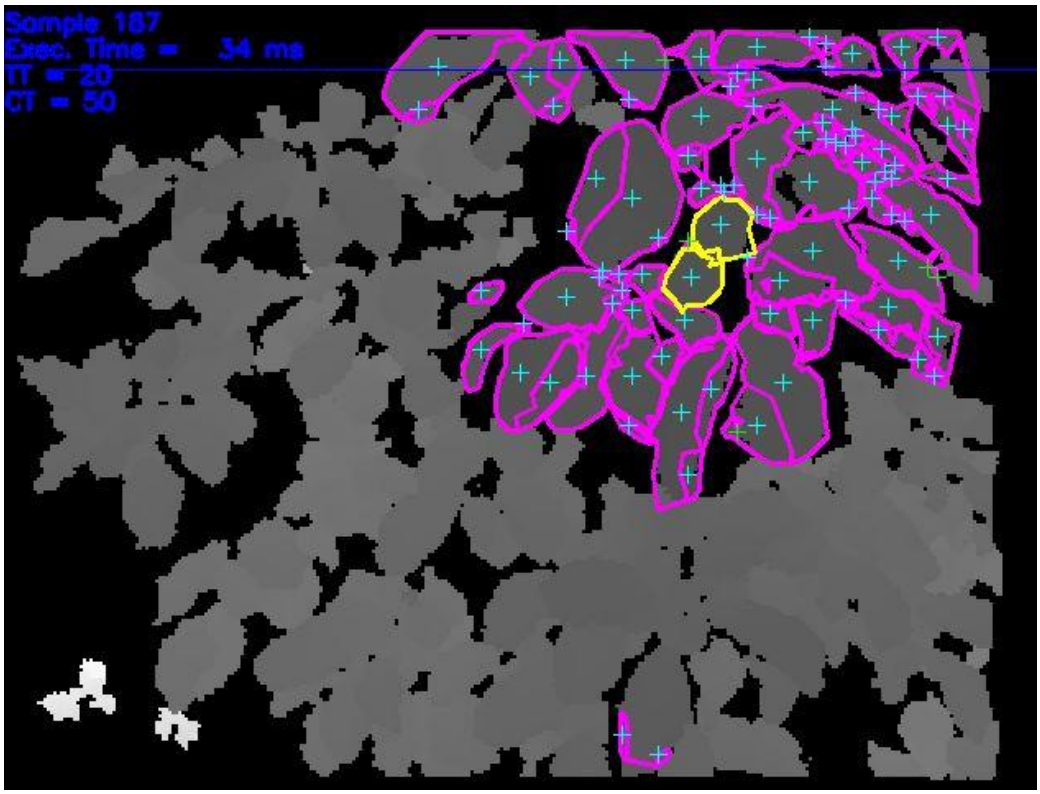
Biblioteka unutar koje se nalazi ovaj rad omogućuje i 3D prikaz slika sa segmentima(Slika 6.1), svaki segment je označen drugom bojom.

Slika 6.1: Primjer 3D prikaza segmentacije(*lijevo*) i njima odgovarajuće RGB slike(*desno*)

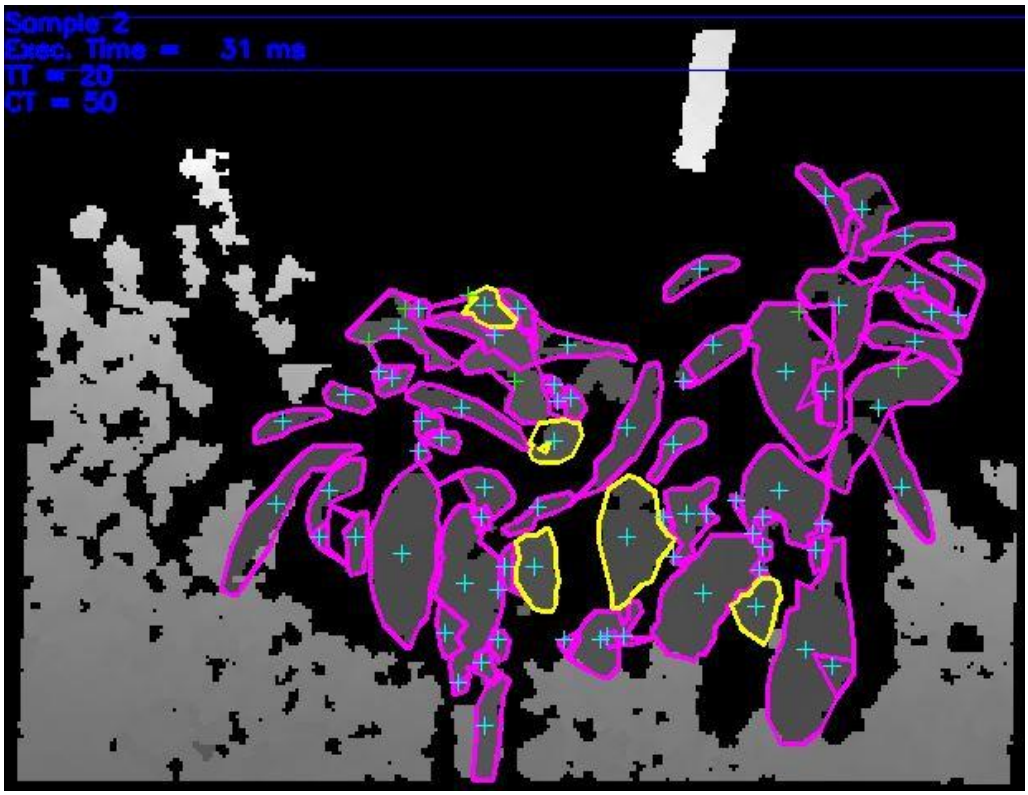
Skup slika je podijeljen na dva dijela, prvi služi za učenje i sadrži 1478 nasumično odabranih slika. Preostalih 1496 slika služi za evaluaciju programa. U skupu za treniranje ima 184766 približno konveksna površina, segmenata, od kojih 7846 pripada voćkama, a preostale pripadaju objektima poput lišća ili grana.

Slike 6.2 i 6.3 prikazuju mjesta ključnih točki segmenata. Ključna točka je središnja točka oblaka, računajući po u i v osi sa 2D slike, označena plavim križićem. Ako ta točka na slici nema svog

predstavnika u 3D oblaku točaka, tada je njoj najbliža točka koja ima svog predstavnika postavljena kao ključna(označena zelenim križićem).



Slika 6.2: Prikaz ključnih točaka na 2D slici prikaza udaljenosti, šljiva



Slika:6.3: Prikaz ključnih točaka na dubinskoj slici, nektarina

Test se provodi u dvije iteracije, u prvoj testiranje se obavlja samo s podacima o obliku površine segmenata, dok u drugoj dodatno postoje informacije i o bojama 3D točaka oblaka. Radi usporedbe složenosti i učinkovitosti deskriptora u svakoj iteraciji provodi se klasifikacija pomoću oba deskriptora. Rezultati su prikazani u kontigencijskoj tablici(engl. *contigency table*), koju prikazuje slika 6.5 i stopom pogreške koja se izražava pomoću osjetljivosti *R*, preciznosti *P* i točnosti testa *A*. Indeks *P* uz varijable predstavlja pozitivnu klasu to jest klasu voća, a *N* predstavlja negativnu klasu, točnije sve ostale objekte koji se pojavljuju na sceni i koji nisu od izravnog interesa. Nadalje indeks T predstavlja ispravno klasificirane segmente, dok indeks F predstavlja netočno klasificirane segmente. Stopa pogreške se računa prema sljedećim formulama[9]:

$$P_P = \frac{TP}{TP + FP} \tag{6-1}$$

$$R_P = \frac{TP}{TP + FN} \tag{6-2}$$

$$A_P = A_N = \frac{TP + TN}{TP + TN + FP + FN} \tag{6-3}$$

$$P_N = \frac{TN}{TN + FN} \tag{6-4}$$

$$R_N = \frac{TN}{TN + FP} \tag{6-5}$$

gdje je *TP* broj segmenata koji su pravilno klasificirani kao voće, *FP* broj segmenata koji su krivo klasificirani kao voće, *FN* broj segmenata koji su krivo klasificirani kao ostali objekti i *FP* broj segmenata koji su pravilno klasificirani kao ostali objekti.

		PREDVIĐENA KLASA	
		Pozitivna klasa	Negativna klasa
STVARNA KLASA	Pozitivna klasa	Stvarno pozitivni rezultati (TP)	Lažno negativni rezultati (FN)
	Negativna klasa	Lažno pozitivni rezultati (FP)	Stvarno negativni rezultati (FP)

Slika 6.5: Primjer kontigencijske tablice

Tablica 6.1: Kontigencijska tablica klasifikacije histogramom distribucije normala

		Predviđena klasa segmenta		
		Voće	Ostali objekti	UKUPNO:
Stvarna klasa segmenta	Voće	176	8384	8560
	Ostali objekti	191181	273	191454
UKUPNO:		191357	8657	200014

Iz tablice 6.1 moguće je izračunati prema formulama 6-1 do 6-5 preciznost, osjetljivost i točnost, koji su prikazani u Tablici 6.2:

Tablica 6.2: Stopa pogrešaka klasifikacije na temelju histograma distribucije normala

	Preciznost *P*	**Osjetljivost *R***	**Točnost *A***
Voće	39,1982%	2,05607%	95,6718%
Ostali objekti	99,8574%	99,8582%	95,6718%

Podatci iz tablice 6.2 prikazuju izrazito lošu preciznost i osjetljivost kod klasifikacije voća, i bez usporedbe moguće je zaključiti da klasifikacija na temelju ovog deskriptora nije pogodna za stvarnu primjenu.

Tablica 6.3: Kontigencijska tablica klasifikacije SHOT deskriptorom

		Predviđena klasa segmenta		
		Voće	Ostali objekti	UKUPNO:
Stvarna klasa segmenta	Voće	7216	1344	8560
	Ostali objekti	189140	2314	191454
UKUPNO:		196356	3658	200014

Izračunom stope pogrešaka iz tablice 6.3 pomoću formula 6-1 do 6-5 slijede rezultati za klasifikaciju FLANN-om na temelju SHOT deskriptora:

Tablica 6.4: Stopa pogrešaka klasifikacije na temelju SHOT deskriptora

	Preciznost *P*	**Osjetljivost *R***	**Točnost *A***
Voće	75,7188%	84,2991%	98,1711%
Ostali objekti	99,2944%	98,7914%	98,1711%

Usporedbom tablica 6.2 i 6.4 može se zaključiti da kompleksnost i detaljnost opisa segmenta povećava osjetljivost za 41 put, što se očituje pravilnim klasificiranjem oko 84,3% od ukupnog boja voćki. Također povećava preciznost za 1,93 puta, što se pak očituje time što je oko 75,72% klasifikacije voća bilo ispravno.

Prijašnji rezultati koristili su samo opis 3D geometrije segmenata prilikom klasifikacije, no krajnji cilj je testiranje na RGB-D slikama, stoga su deskriptorima dodane informacije i o bojama točaka oblaka. Tablice 6.5 i 6.6 prikazuju rezultate za metodu opisivanja segmenata histogramom distribucije noramla uz dodane informacije o bojama.

Tablica 6.5: Kontigencijska tablica histograma distribucije normala sa srednjom vrijednosti boja

		Predviđena klasa segmenta		
		Voće	Ostali objekti	UKUPNO:
Stvarna klasa segmenta	Voće	182	8378	8560
	Ostali objekti	191174	280	191454
UKUPNO:		191356	8658	200014

Prema prije navedenim formulama izračun stope pogrešaka daje:

Tablica 6.6: Stopa pogrešaka histograma distribucije normala sa bojom

	Preciznost *P*	**Osjetljivost *R***	**Točnost *A***
Voće	39,3939%	2,0308%	95,5039%
Ostali objekti	95,6329%	99,8546%	95,5039%

Tablice 6.7 i 6.8 prikazuje rezultate klasifikacije korištenjem SHOT deskriptora za RGB-D podatke.

Tablica 6.7: Kontigencijska tablica klasifikacije na temelju SHOT-a za RGB-D

		Predviđena klasa segmenta		
		Voće	Ostali objekti	UKUPNO:
Stvarna klasa segmenta	Voće	7320	1240	8560
	Ostali objekti	189240	2214	191454
UKUPNO:		196560	3454	200014

Tablica 6.8: Stopa pogrešaka SHOT-a za RGB-D podatke

	Preciznost *P*	**Osjetljivost *R***	**Točnost *A***
Voće	76,7778%	85,514%	98,2731%
Ostali objekti	99,349%	98,846%	98,2731%

Usporedbom rezultata kod korištenja dekriptora bez informacije o boji sa deskriptorima u kojima je ta informacija sadržana vidljivo je da su rezultati vrlo slični, točnije informacije o boji nisu imale veći utjecaj na uspješnost klasifikacija. Kod histograma distribucije normala promijene su bila unutar jednog postotka, i to u negativnom smislu jer su se osjetljivost i točnost smanjile. Kod SHOT deskriptora promijene također nisu značajnije, povećanje kod preciznosti i osjetljivosti je oko jedan posto. Istovremeno u SHOT deskriptoru informacije o boji 3,82 puta povećavaju deskriptor što značajno utječe na vrijeme obrade podataka, osobito zbog toga što pri svakoj klasifikaciji deskriptora FLANN mora pretraživati skup za treniranje koji je velik i nije organiziran.

7. ZAKLJUČAK

U ovom je radu izrađen, predstavljen i testiran program za klasifikaciju voća na temelju dva deskriptora, histograma distribucije normala i SHOT deskriptora. Kao teorijska podloga objašnjene su ključne komponente programa. Potom je objašnjen tijek programa i njegove sastavne cjeline. Posljednji dio rada opisuje skup nad kojim je vršeno treniranje i testiranje, te usporedbu rezultata dviju korištenih deskriptora.

Krajnji cilj rada je bio testiranjem provjeriti može li se koristiti trenutno dostupna tehnologija i programska rješenja za automatizaciju branja voća. Zaključak donesen na temelju rezultata jest da korištenje histograma distribucije normala u stvarnosti nije moguć, no rezultati korištenja SHOT deskriptora pokazuju da postoji mogućnost za njegovu uporabu. Također zaključak je da informacije o boji malo poboljšavaju detekciju, ali bitno povećavaju vrijeme izvođenja programa. Proširenjem ovog rada na testiranje više vrsta deskriptora, trenutno dostupnih za javnost, moglo bi rezultirati pronalaskom optimalnog deskriptora za primjenu u voćarstvu. Također, provođenjem usporednog testa SHOT deskriptora s nekoliko različitih postavki ustanovila bi se mogućnost optimizacije tog deskriptora, jer ovaj rad je koristio samo jedne postavke.

LITERATURA

[1] RICHARD HOFFMAN, ANIL K. JAIN - Segmentation and Classification of Range Images http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4767955&isnumber=4767950

[2] Robert Cupec Emmanule K. Nyarko, Damir Filko - Fast 2.5D Mesh Segmentation to Approximately Convex Surfaces, 2011

[3] Federico Tombari, Samuele Salti, Luigi Di Stefano - SHOT: Unique Signatures of Histograms for Surface and Texture Description, https://www.researchgate.net/publication/262111100_SHOT_Unique_Signatures_of_Histograms_for_Surface_and_Texture_Description

[4] Andreas Koschan, Mongi Abidi - Digital Color Image Proccessing https://goo.gl/cwzwxH

[5] Federico Tombari, Samuele Salti i Luigi Di Stefano - Unique Signatures of Histograms for Local Surface Description https://www.vision.deis.unibo.it/fede/papers/eccv10.pdf

[6] Federico Tombari, Samuele Salti, Luigi Di Stefano - SHOT: Unique Signatures of Histograms for Surface and Texture Description, https://www.researchgate.net/publication/262111100_SHOT_Unique_Signatures_of_Histograms_for_Surface_and_Texture_Description

[7] dr. sc. Maja Strgar Kurečić - Kontrola boja - od percepcije do mjerenja http://repro.grf.unizg.hr/media/download_gallery/OSNOVE%20O%20BOJI.pdf

[8] Marius Muja, David G. Lowe - FAST APPROXIMATE NEAREST NEIGHBORS WITH AUTOMATIC ALGORITHM CONFIGURATION http://www.cs.ubc.ca/research/flann/uploads/FLANN/flann_visapp09.pdf

[9] https://www.slideshare.net/sriphaew/dbm630-lecture08, pristup 11. srpnja 2017.

[10] Robert Cupec, Damir Filko, Ivan Vidović, Emmanuel Karlo Nyarko, Željko Hocenski - Point Cloud Segmentation to Approximately Convex Surfaces for Fruit Recognition, Proceedings of The Croatian Computer Vision Workshop, CCVW 2014, 2014. pp. 56-61.

SAŽETAK

Razvijen je program za prepoznavanje voća na RGB-D slikama. Cilj programa je testiranje trenutno dostupne tehnologije u primjeni automatizacije branja voća, a rezultati testiranja pokazali su da trenutna tehnologija ima potencijal za korištenje u automatskom branju voća. Ovaj rad se fokusirao na testiranje učinkovitosti dva deskriptora: histograma distribucije normala i SHOT deskriptora. Deskriptori se izračunavaju za svaki segment dobiven segmentacijom RGB-D slike na približno konveksne površine. Klasifikacija segmenata na voće i ostale objekte, koji ne predstavljaju voće, ostvarena je metodom k-najbližih susjeda implementiranom pomoću FLANN biblioteke. Učinkovitosti dviju razvijenih metoda ispitane su na skupu od 2974 RGB-D slika.

Ključne riječi: RGB-D, klasifikacija, segmentacija, deskriptor, detekcija voća, SHOT, histogram distribucije normala

ABSTRACT

A program for fruit recognition in RGB-D images is developed. The programs goal was to investigate the practical applicability of the technologies available today for fruit picking automation. Test results have shown that this technology have potential for use in automated fruit picking. The focus of this work is on comparison of two descriptors: normal distribution histogram and SHOT descriptor. The descriptors are calculated for every segment provided by segmentation of RGB-D image to approximately convex surfaces. Classification of the obtained segment on fruits and objects that do not represent fruits was accomplished with the k-nearest neighbors classifier implemented using FLANN library. The efficiency of two the developed methods was tested on a set of 2974 RGB-D images.

Keywords: RGB-D, classification, segmentation, descriptor, fruit detection, SHOT, normal distribution histogram

ŽIVOTOPIS

Kristijan Radočaj rođen je u Virovitici 30. rujna 1991. godine. Nakon završene osnovne škole(OŠ Eugena Kumičića, Slatina) upisuje srednju školu Marka Marulića Slatina, smjer elektrotehničar. Godine 2010. maturira sa vrlo dobrim uspjehom, nakon čega upisuje trogodišnji stručni studij, smjer informatike, na Elektrotehničkom fakultetu u Osijeku. Nakon završetka stručnog studija upisuje razlikovnu godinu, a 2014. godine upisuje sveučilišni diplomski studij, smjer Procesno računarstvo.

Kristijan Radočaj