SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINAR

Robusna estimacija konsenzusom slučajnih uzoraka (RANSAC)

Andrea Žabčić

Voditelj: *Siniša Šegvić*

Zagreb, travanj, 2009

Sadržaj

1. Uvod

Čest slučaj u računalnom vidu je procjenjivanje analitičkog modela, kao što su krivulja i površina, na temelju podataka koji sadrže dvije vrste grešaka: greške mjerenja i asocijacijske greške. Greške mjerenja se često mogu modelirati kao male, normalno distribuirane devijacije od modela kojeg želimo procijeniti. S druge strane neki podaci uopće ne pripadaju populaciji koju želimo modelirati uslijed grube pogreške u fazi odabira podataka. Takve greške zovemo asocijacijske greške. Te greške su proizvoljne i općenito se ne mogu prigušiti konvencionalnim metodama procjene modela. Kažemo da su takvi podaci vanpopulacijski, te su skupu pridruženi asocijacijskom greškom.

Metoda estimacije konsenzusom slučajnih uzoraka (RANSAC) može tolerirati vanpopulacijske podatke, te model procjenjuje samo na temelju podataka koji su u skladu sa pretpostavljenim modelom. Na taj način dobivamo preciznije modele, jer onemogućavamo vanpopulacijskim podacima da pokvare rezultate.

U ovom radu prvo ćemo unutar poglavlja 2 objasniti glavna svojstva algoritma RANSAC, nakon toga ćemo se detaljnije posvetiti načinu rada algoritma u poglavlju 3, te njegovoj implementaciji u programskom okruženju Matlab u poglavlju 4. Promotriti ćemo konkretne primjene algoritma u poglavlju 5, te navesti kratki zaključak rada.

2. Konsenzus slučajnih uzoraka

Konsenzus slučajnih uzoraka (RANSAC) je iterativna metoda za procjenu parametara matematičkog modela iz skupa podataka. Osnovna pretpostavka je da se podaci sastoje od populacijskih i vanpopulacijskih podataka. Populacijski podaci su podaci čija se distribucija može objasniti modelom, dok su vanpopulacijski podaci koji se ne mogu ukomponirati u model. Algoritam je prvi puta objavljen od strane Fischlera i Bollesa 1981. godine [2].

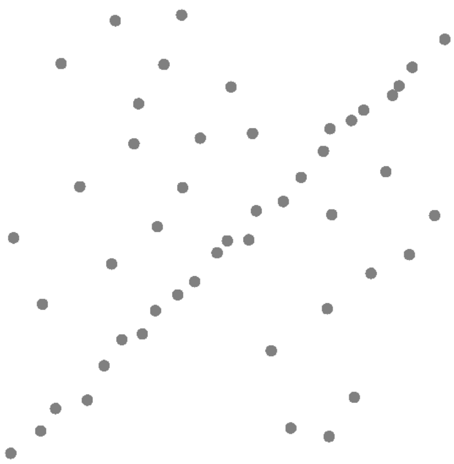
Konvencionalne tehnike počinju sa što više podataka te pokušavaju eliminirati vanpopulacijske podatke. Algoritam RANSAC s druge strane koristi najmanji mogući početni skup podatka i povećava ga podacima koji pripadaju modelu.

Glavna ideja je da se slučajnim odabirom minimalnog uzorka hipotetizira traženi model. Hipoteze se zatim evaluiraju nad svim elementima ulaznog skupa podataka. Algoritam vraća najbolju hipotezu.

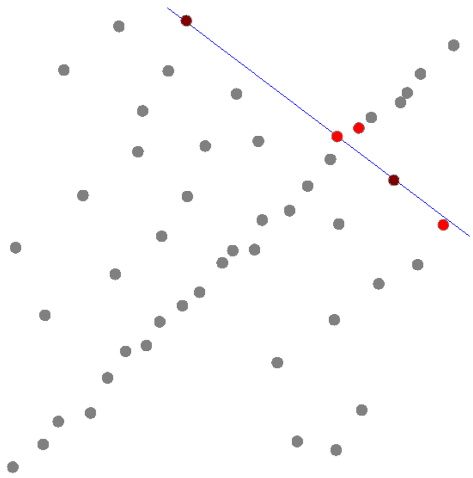
Formalno ćemo RANSAC definirati na slijedeći način [2]:

> Pretpostavimo da želimo estimirati model koji zahtjeva minimalno n vektora za definiranje modela, i skup vektora P takav da je |P|>n. Slučajnim odabirom izabiremo podskup S1 iz P sa n vektora i na temelju njih definiramo model M1. Koristimo predviđeni model M1 da bismo odredili podskup S1* skupa točaka P koje odgovaraju modelu M1 sa odstupanjem manjim od zadanog. Skup S1* zovemo konsenzusnom grupom od S1.
>
> Ako je veličina S1* skupa veća od nekog praga t, pronašli smo do sada najbolju konsenzusnu grupu. Korištenjem tih podatka konsenzusne grupe nekom drugom metodom procijenimo model, koji će sada biti optimalan za taj skup.
>
> Ako je veličina S1* manja od t, slučajnim odabirom izaberi novi podskup S2 i ponovi gore opisani postupak. Nakon unaprijed definiranog broja ponavljanja zaustavljamo rad algoritma te za rezultat uzimamo najbolju hipotezu.

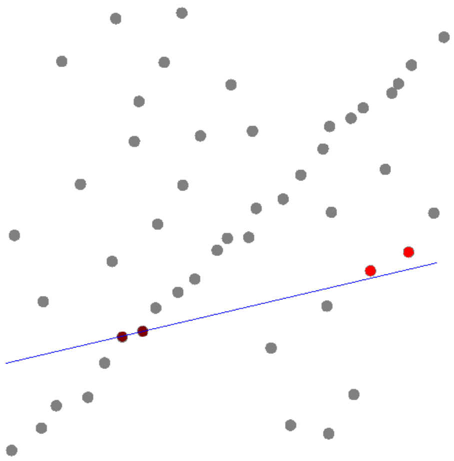
Pogledati ćemo RANSAC algoritam na nekoliko konkretnih primjera. Krenuti ćemo on najjednostavnijeg primjera: modeliranja pravca. Za modeliranje pravca postupak počinje sa odabirom dviju točaka kroz koje se zatim provlači pravac. Računaju se udaljenosti svih točaka od pravca, te ukoliko je točka dovoljno blizu pravca ona se dodaje u konsenzus grupu. Taj postupak ponavljamo više puta sa različitim početnim točkama, te pronalazimo pravac koji sadrži najveću konsenzusnu grupu. Na slici 1 je prikazan skup podataka iz kojih pokušavamo povući pravac, dok su na slici 2 i 3 prikazani primjeri iteracija algoritma. Slučajnim odabirom su izabrane dvije točke te se evaluirala hipoteza nad podacima. Na slici 4 je prikazan konačan rezultat algoritma, odnosno najbolji pronađeni model.



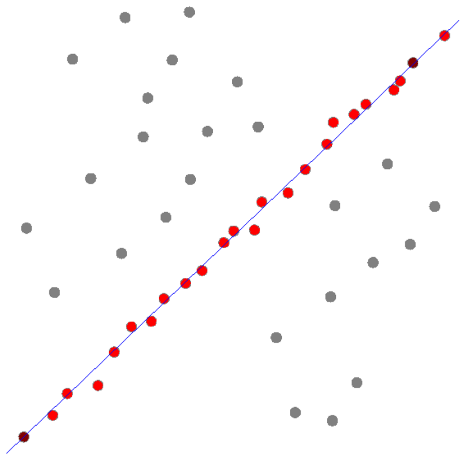
Slika 1: skup podataka



Slika 2: prva iteracija



Slika 3: druga iteracija



Slika 4: rezultat algoritma

Slično kao i u slučaju pravca ako bi nam ciljani model bila kružnica za početak bi nam bile potrebne tri točke. Na temelju te tri točke izračunali bismo središte i polumjer kružnice, te dalje pronašli točke koje su dovoljno blizu kruga. Ukoliko bi ciljani model bila ravnina također bi nam bile dovoljne tri točke za definiciju ravnine.

3. Algoritam

3.1 Pseudokôd algoritma

Kako bismo bolje prikazali način na koji radi robusna estimacija konsenzusom slučajnih uzoraka promotrit ćemo detaljnije konkretni algoritam.

Pseudokod [1]:

```
Ulazni podaci:
      podaci – skup podataka iz kojih želimo procijeniti parametre
              matematičkog modela
      model – model koji se može procijeniti iz podataka
      n – minimalni broj podataka potreban za procjenu modela
      k – maksimalni broj iteracija dozvoljen u algoritmu
      er – maksimalno odstupanje podatka od modela
      t – broj bliskih podataka nužan za dokazivanje da promatrani model
          dobro odgovara podacima
```

Glavni ulazni podaci algoritma su `podaci` u kojem se nalaze podaci iz kojih želimo procijeniti parametre, te ciljani model koji želimo procijeniti iz podataka. Još su nam potrebni neki pomoćni parametri kao što su: minimalni broj podataka potrebnih za procjenu modela, maksimalni broj iteracija u algoritmu, maksimalno odstupanje podataka od modela, te broj bliskih podataka nužan za dokazivanje da promatrani model dobro odgovara podacima. Vrijednosti tih podataka, osim vrijednosti minimalnog broja podataka potrebnih za procjenu modela, izabiremo sami. Moramo ih pažljivo odrediti kako bi algoritam bio što točniji i što brži.

```
Izlazni podaci:
      najbolji_model – parametri modela koji najbolje odgovaraju podacima
      najbolja_konsenzusna_grupa – podaci koji pripadaju modelu
```

Kao izlaz algoritma dobiti ćemo konkretni model procijenjen iz podataka i podatke koji tvore taj model.

```
iteracija := 0
najbolji_model := nil
najbolja_konsenzusna_grupa:= nil
```

Na početku inicijaliziramo sve lokalne varijable na početne, neutralne vrijednosti. `Najbolji_model` i `najbolju_konsenzusnu_grupu` inicijaliziramo tako da ukoliko algoritam ne pronađe niti jedan model vratit će neutralne vrijednosti na temelju kojih ćemo znati da nije pronađen niti jedan model.

```
dok je iteracija < k
```

Postupak ponavljamo k iteracija, gdje nam je varijabla k maksimalan broj iteracija algoritma. Ukoliko u k iteracija algoritma ne pronađemo niti jedan model, skup podataka koji promatramo nije dobar za taj konkretni model kojeg želimo procijeniti.

```
    uzorak := n slučajno odabranih podataka iz podaci
    hipoteza := parametri modela procijenjenog iz uzorka
    konsenzusna_grupa := uzorak
```

U svakoj iteraciji iz skupa `podaci` izabiremo slučajnim odabirom `n` podataka, gdje je n minimalan broj podataka potreban za procjenu modela. Npr. za pravac su to dvije točke, a za kružnicu tri. Dodatno se u ovom trenutku za neke modele još mora i provjeriti ispravnost podataka, tj. da li je raspored točaka dobar za procjenu modela. Ukoliko je model npr. ploha za procjenu modela potrebne su nam tri točke, ali te tri točke moraju biti kolinearne. Točke su kolinearne ako se kroz njih može provući ploha, odnosno ako se ne nalaze na jednom pravcu.

Iz odabranih podataka procjenjujemo traženi model, te u konsenzusnu grupu stavljamo tih n slučajno odabranih podataka. Kasnije ćemo tu grupu proširiti sa drugim podacima koji pripadaju ovome modelu.

```
    za svaku točku iz podaci koja nije u uzorku
        ako točka pristaje u hipotezu sa greškom manjom od er
                dodaj točku u konsenzusnu_grupu
```

Dalje uzimamo sve točke iz početnog skupa izuzev onih koje smo prethodno izabrali slučajnim odabirom i stavili u `konsenzusnu_grupu`. Za svaku točku provjeravamo da li pristaje u potencijalni model sa odstupanjem manjim od t. Ukoliko pristaje u model dodajemo ju u `konsenzusnu_grupu`.

```
    ako je broj elemenata u konsenzusnoj_grupi > t
```

Ukoliko je broj elemenata u `konsenzusnoj_grupi` veći od t, tj. veći od minimalnog broja podataka koji nam je potreban za procjenu modela, pronašli smo dobar model. Sada je potrebno provjeriti da li je promatrani model bolji od prethodno pronađenog modela.

```
    ako je broj elemenata u konsenzusnoj_grupi > broja elemenata u
    najboljoj_konsenzusnoj_grupi
        bolji_model := parametri modela koji pristaju svim točkama u
        konsenzusnoj_grupi
        najbolji_model := bolji_model
        najbolja_konsenzusna_grupa := konsenzusna_grupa
```

Ukoliko je trenutno promatrana `konsenzusna_grupa` veća od prethodno pronađene grupe ovaj model je najbolji do sad pronađeni, te ga moramo spremiti. Najprije ćemo na temelju podataka iz `konsenzusne_grupe` ponovo izračunati parametre modela, koji će biti sličan početnom modelu ali vjerojatno ne i identičan. Na početku iteracije izabrali smo nekoliko točaka koje su definirale neki model, te smo zatim u grupu stavili sve točke koje su unutar odstupanja od toga modela. No možda za tu konsenzusnu grupu postoji i bolji model od onog izabranog na početku, te ćemo ovdje taj model izračunati.

U `najbolji_model` i `najbolju_konsenzusnu_grupu` spremamo podatke o promatranom modelu.

```
    povecaj iteraciju

vrati najbolji_model, najbolja_konsenzusna_grupa
```

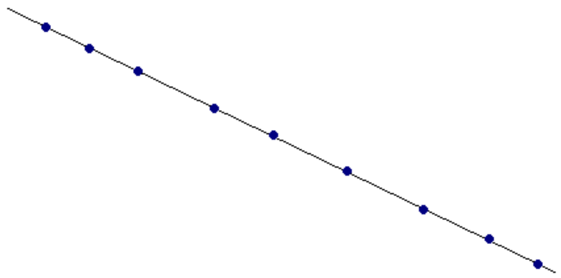
Algoritam na kraju vraća najbolji model kojeg je procijenio iz podataka i pripadajuću konsenzusnu grupu.

3.2 Određivanje vrijednosti parametara

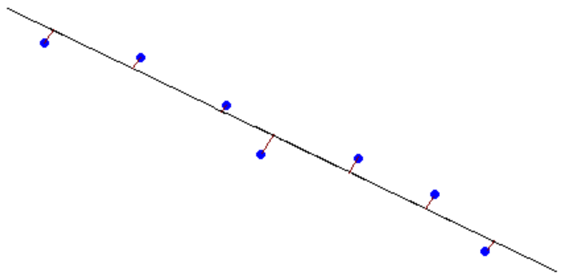
U postupku smo spomenuli tri parametra: maksimalno odstupanje podataka od modela (er), maksimalni broj iteracija (k) i broj bliskih podataka nužan za dokazivanje da promatrani model odgovara podacima, odnosno minimalni broj članova konsenzus grupe (t). Potrebno je na neki način odrediti konkretne vrijednosti tih parametara, tako da postignemo što točnije rezultate.

Prvo ćemo razmotriti iznos maksimalnog odstupanja podataka. Ukoliko je model jednostavna funkcija može se često analitički odrediti vrijednost tog parametra, no za složenije funkcije često takav način nije dovoljan. U takvim slučajevima maksimalno dopušteno odstupanje ćemo procijeniti eksperimentalno. Uzorke odstupanja možemo dobiti unošenjem asocijacijskog šuma, izračunavanjem modela i mjerenjem izazvanih odstupanja. Na slikama 5 i 6 je ilustriran taj postupak. Na slici 5 su prikazani podaci bez šuma, dok je na slici 6 uveden šum.

Maksimalno odstupanje se tada može postaviti za jednu ili dvije standardne devijacije više od izmjerene prosječne vrijednosti.



Slika 5: uzorak bez šuma



Slika 6: uzorak sa unešenim šumom

Za svaki skup podataka će mjereno odstupanje biti drugačije, te bi maksimalno odstupanje trebalo biti drugačije za svaki skup. No, te varijacije u odstupanjima su uglavnom relativno male u usporedbi sa veličinom odstupanja nekog vanpopulacijskog podatka, te je dovoljna jedna vrijednost maksimalnog odstupanja za sve skupove podataka.

Slijedeći parametar je maksimalni broj iteracija. Potrebno ga je odrediti tako da bude dovoljno veliki da omogući pronalaženje odgovarajućeg modela, ali opet ne preveliki, da beskorisno ne troši vrijeme u iteracijama. Odluka o prestanku uzimanja novih podskupova podataka može biti temeljena na očekivanom broju pokušaja (k) do pronalaženja prihvatljivog modela, tj. do dobrog odabira n točaka iz skupa, gdje je n minimalni broj točaka potreban za procjenu parametara modela. Neka je w vjerojatnost da je bilo koja izabrana vrijednost unutar maksimalnog odstupanja od modela, odnosno w je postotak populacijskih podataka. Tada imamo (1), gdje je E(k) očekivana vrijednost k.

$$E(k) = w^n + 2*(1-w^n)*w^n + 3*(1-w^n)^2*w^n + \ldots + i*(1-w^n)^{i-1}*w^n + \ldots \qquad (1)$$

Dalje slijedi (2):

$$E(k) = w^n (1 + 2 * (1 - w^n) + 3 * (1 - w^n)^2 + ... + i * (1 - w^n)^{i-1} + ...) \tag{2}$$

Kada izraz (1-w^n) zamijenimo sa izrazom *a* iz izraza (3) lagano se prepoznaje da je to derivacija geometrijskog niza. Formula za geometrijski niz je dana izrazom (4).

$$E(k) = w^n (1 + 2 * a + 3 * a^2 + ... + i * a^{i-1} + ...) \tag{3}$$

$$\frac{a}{1-a} = a + a^2 + a^3 + ... + a^i \tag{4}$$

Deriviranjem toga izraza dobivamo(5):

$$\frac{1}{(1-a)^2} = 1 + 2 * a \quad + 3 * a^2 + ... + i * a^{i-1} \tag{5}$$

Korištenjem toga izraz E(k) je zatim jednak (6):

$$E(k) = \frac{w^n}{(1-(1-w^n))^2} = \frac{1}{w^n} = w^{-n} \tag{6}$$

Za svaki slučaj ćemo za broj iteracija uzeti dva ili tri puta veći broj od dobivenog E(k).

Možemo tu vrijednost izračunati i na malo drugačiji način. Želimo li osigurati sa vjerojatnost z da će barem jedan on naših slučajnih odabira biti skup od n točaka bez grešaka, tada moramo očekivati da će nam biti potrebno barem k odabira, gdje je (7):

$$(1 - w^n)^k = (1 - z) \tag{7}$$

Odnosno (8):

$$k = [\log(1 - z)] / [\log(1 - w^n)] \tag{8}$$

Treći parametar je minimalni broj članova konsenzus grupe (t). On nam služi kao osnova za odluku da li je podskup podataka koji smo pronašli dovoljno velik za procjenu modela. Zato t mora biti izabran dovoljno veliki da zadovolji dva cilja: da je pronađen točan model iz podataka, i da je pronađen dovoljno veliki broj točaka koji zadovoljava potrebe završne procedure izglađivanja (koja izračunava poboljšane procjene modelskih parametara).

Da bi se osigurali od mogućnosti da je konačna konsenzus grupa kompatibilna sa pogrešnim modelom, uz pretpostavku da je y vjerojatnost da je bilo koja točka unutar prihvatljivog odstupanja od pogrešnog modela, htjeli bi da je y^{t-n} mala vrijednost. Iako ne postoji ni jedna metoda za precizno određivanje y, logično je zaključiti da je y manji od w (vjerojatnost da je bilo koja točka unutar dopuštenog odstupanja od točnog modela). Ukoliko pretpostavimo da je $y<0.5$, vrijednost t-n jednaka 5 će osigurati 95% sigurnost da se neće dogoditi poklapanje sa pogrešnim modelom.

4. Implementacija u programskom okruženju Matlab

Koristeći implementaciju algoritma za slučaj pravca u programskom okruženju Matlab testirat ćemo rezultate algoritma mijenjajući neke od parametara algoritma kao što su: broj iteracija, odnos populacijskih i vanpopulacijskih podataka i drugo.

Postupak ćemo krenuti od generirane idealne skupine podataka, tj od podataka koji se svi nalaze na nekom pravcu bez odstupanja. Zatim ćemo na te podatke primijeniti neki šum da bi dobili realniju situaciju, te dodali određeni postotak vanpopulacijski podataka. Primijeniti ćemo algoritam na oba pravca, onaj idealni i novonastali sa šumom, te mjerili odstupanje promijenjenog pravca od onog izvornog. Uzmemo li implicitne jednadžbe pravca koje su oblika (9):

$$ax + by + c = 0 \tag{9}$$

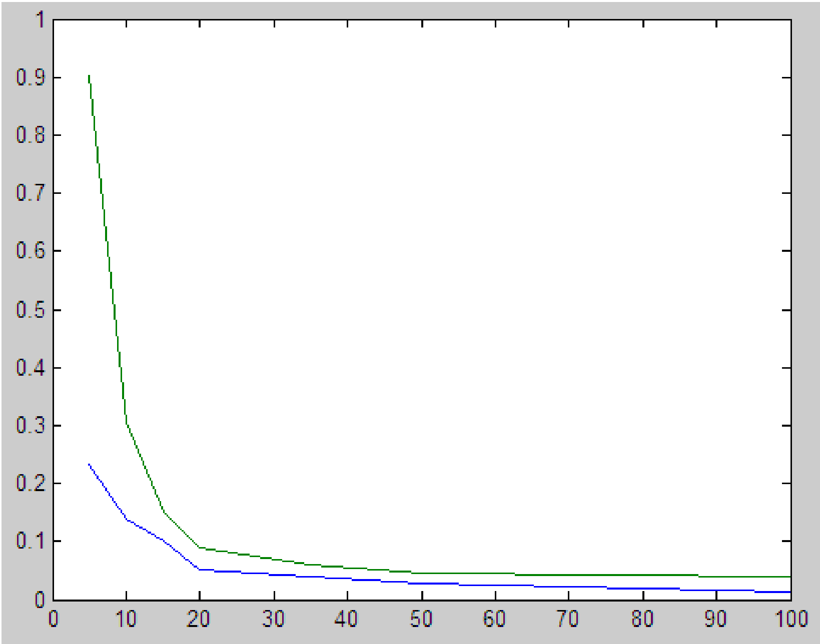
Odstupanje između dva pravca mjerit ćemo slijedećom formulom(10):

$$\delta(p1, p2) = \left\| \begin{bmatrix} a1 \\ b1 \\ c1 \end{bmatrix} - \begin{bmatrix} a2 \\ b2 \\ c2 \end{bmatrix} \right\| \tag{10}$$

Za početak ćemo krenuti od situacije gdje se u podacima nalazi samo 10% vanpopulacijskih podataka. U ostale populacijske podatke unijeti ćemo šum σ. Šum će nam promijeniti podatke tako da njihovu vrijednost smanji ili poveća za proizvoljno broj iz intervala [0, σ]. Na ovako nastalim podacima korištenjem različitih vrijednost σ možemo izračunati koja će biti veličina greške, odnosno odstupanje od originalnih pravaca, s obzirom na izabrani broj iteracija.

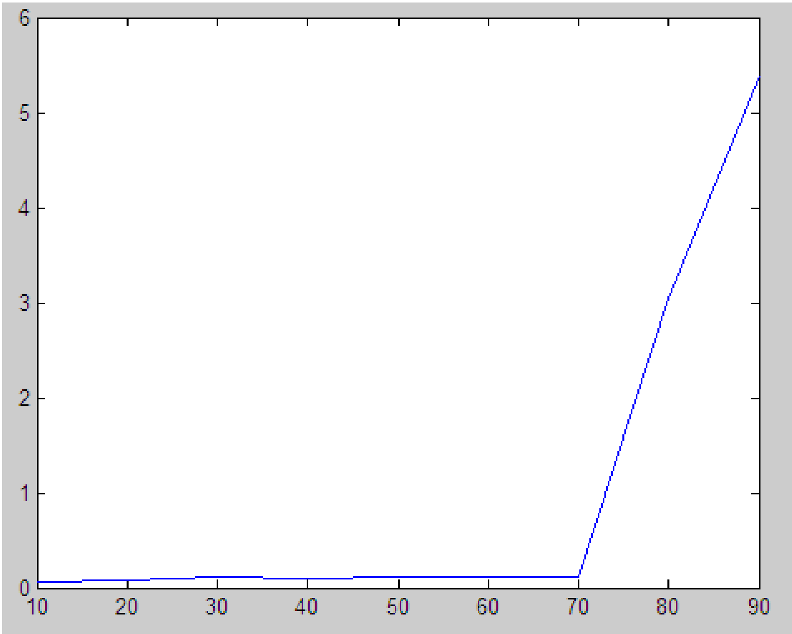
Ukoliko uzmemo skup od 20 točaka, te za parametar maksimalnog odstupanja od modela uzmemo vrijednost 0.5, možemo promotriti kakva će biti greška modela za količinu šuma σ=0.2 i σ =0.5. Na slici 7 je prikazan iznos grešaka modela u odnosu na broj iteracija za ta dva slučaja. Plavom bojom je označen skup podataka sa šumom σ=0.2, dok je zelenom bojom označen skup podataka sa šumom σ =0.5. Sasvim je očekivano da je prosječna greška u drugom slučaju veća, pošto je veća količina odstupanja.

Možemo primijetiti da se iznos greške modela nakon nekog vremena prestaje smanjivati, zato ima smisla za ovaj slučaj od 20 točaka ograničiti broj iteracija na neku vrijednost oko 50.



Slika 7: iznos grešaka modela u odnosu na broj iteracija i iznos šuma

Nadalje, razmotrit ćemo utjecaj količine vanpopulacijskih podataka na rezultat algoritma. Uzet ćemo prethodno izračunati skup sa količinom šuma od σ=0.2, te povećavati broj vanpopulacijskih podataka. Vanpopulacijske podatke ćemo generirati tako da ih uniformnom razdiobom razmjestimo po cijeloj slici. Na slici 8 je prikazan prosječan iznos greške u odnosu na količinu vanpopulacijskih podataka za deset iteracija testiranja algoritma. Ono što možemo primijetiti je da su rezultati relativno dobri do unošenja 70 % vanpopulacijskih podataka. Rezultati ostaju dobri čak i u tom slučaju jer je tih 70% podataka razbacano po cijeloj slici i u većini slučajeva ne uspijevaju tvoriti pravac bolji od izvornog pravca kojeg tvori ostalih 30 % podataka.



Slika 8: iznos prosječne greške modela u odnosu na postotak vanpopulacijskih podataka

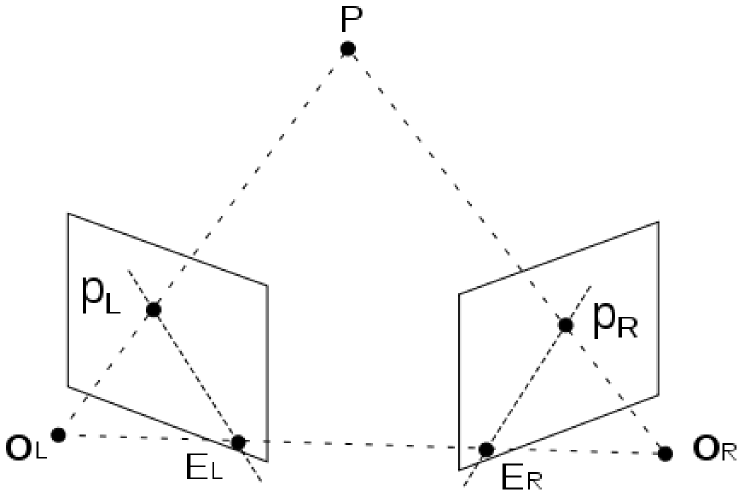
5. Primjene

Zbog svoje robusnosti algoritam RANSAC je našao najveću primjenu u algoritmima računalnog vida gdje je interpretacija bazirana na podacima sklonim asocijacijskim greškama. Ciljevi algoritama računalnog vida su izdvajanje geometrijskih, fotometrijskih i semantičkih informacija sa slike. U svim tim postupcima potrebna nam je neka vrsta procjene parametara za opisivanje intenziteta rubova krivulja, modela kretanja, normala, zakrivljenosti površina i drugih parametara. Upravo za to će nam služiti algoritam RANSAC.

Jedna od primjena algoritma je u uspostavljanju odnosa između elemenata dvije reprezentacije određene scene[5]. RANSAC algoritam nam koristi za prepoznavanje određenih krivulja i oblika koje nakon toga možemo usporediti sa pronađenim krivuljama na drugoj slici te pronaći slične, odnosno jednake. Na slici 9 je prikazan primjer jednog takvog algoritma prepoznavanja oblika na različitim slikama.

Slika 9: prepoznavanje jednakih oblika na različitim slikama iste scene [5]

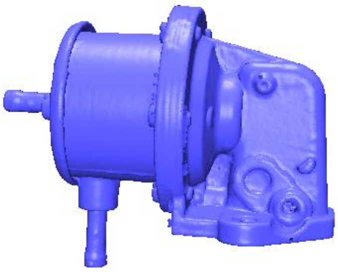
Algoritam koristimo i prilikom estimacije relativne orijentacije tj. epipolarne geometrije. Prepoznajući neke ključne točke na dvije slike istog objekta možemo izračunati pomak i rotaciju kamere. Na slici 10 je pojednostavljeno prikazan slučaj epipolarne geometrije sa dvije kamere Q_L i Q_R dok je P promatrana točka.



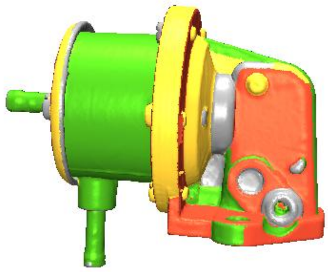
Slika 10:epipolarna geometrija

Za definiranje odnosa između elemenata dvije slike potrebno je izračunati fundamentalnu matricu. Fundamentalna matrica je matrica veličine 3x3 koja točno definira odnos između dvije slike. Za računanje te matrice potrebno je barem sedam skupova (x,x'), gdje je x točka sa prve slike, a x' je pripadajuća točka druge slike. Glavna ideja je: izabere se 7 takvih skupova koji bi mogli predstavljati podudarne točke. Zatim se u odnosu na te točke evaluiraju ostale točke sa obje slike te se izračuna koliko se od njih podudara. Ukoliko se dovoljni postotak njih podudara pronašli smo dobar model preslikavanja s jedne slike na drugu, te korištenjem svih točaka koje se podudaraju nanovo izračunavamo fundamentalnu matricu. U suprotnom izabiremo novih sedam skupova te ponavljamo postupak.

Algoritam RANSAC može se koristit i za prepoznavanje raznih oblika na slici [4]. U tom slučaju koristi se algoritam koji je sposoban prepoznati mnogo različitih tipova primitivnih oblika (plohe,sfere, valjke, stošce...), dok u isto vrijeme zadržava svoje pozitivne značajke: robusnost, jednostavnost i općenitosti. Na slici 11 je prikazan originalni model na kojem želimo prepoznati različite oblike, a na slici 12 su različito obojani prepoznati različiti oblici (površine crveno, valjci zeleno, sfere žuto i torusi sivo). Ovaj algoritam radi tako da izabire iz skupa podataka slučajnim odabirom minimalni skup podataka, na temelju kojih se tada stvaraju kandidati za sve oblike koje tražimo. Nakon toga primijenimo algoritam RANSAC te se detektira onaj oblik koji je uključio najveći broj točaka. Taj skup točaka koji tvori pronađeni oblik se izuzima iz daljnjeg razmatranja.



Slika 11: originalni model [4]



Slika 12: obojano po pronađenim oblicima [4]

Algoritam se koristi i kao pomoć u detekciji prometnog traka na slijedu slika snimljenih iz vozila u pokretu. Ovdje nam koristi za modeliranje prometnih linija. Za model se može uzeti pravac pošto zakrivljenje prometnih linija nije toliko oštro, te se i pravcem uglavnom obuhvate svi dijelovi linije traka.

Navedene primjene su samo neke od mnoštva primjena RANSAC algoritma u analizi slika.

6. Zaključak

Glavna prednost RANSAC algoritma je njegova mogućnost robusnog procjenjivanja parametara modela. On može procijeniti parametre sa visokim stupnjem točnosti čak i kada se u podacima nalazi znatna količina vanpopulacijskih podataka. U većini slučajevima prepoznat će i parametre modela koji su građeni od manje od pola podataka. No zbog toga se može dogoditi da se kao model detektiraju i male, slučajne strukture.

Prilikom realizacije algoritma potrebno je, radi povećanja točnosti i ubrzanja algoritma, posebnu pažnju posvetiti određivanju parametara algoritma. Jedan od parametara je broj iteracija algoritma. Ukoliko se izabere premali broj iteracija pojavljuje se mogućnost da pronađeni model nije najoptimalniji. Zato se broj iteracija mora prilagoditi količini podataka koju želimo evaluirati.

Drugi važan parametar algoritma je iznos maksimalnog odstupanja od modela. Njegovu vrijednost je potrebno izabrati sukladno sa očekivanim vrijednostima odstupanja podataka. Svakako ako se on postavi na premalu vrijednost bi se moglo dogodit da algoritam ne radi na odgovarajući način, te da ne pronalazi traženi model, pošto je većina odstupanja podataka veća od toga praga.

Zbog svoje robusnosti estimacija konsenzusom slučajnih uzoraka se uglavnom koristi kao pomoć u algoritmima računalnoga vida, gdje su podaci skloni asocijacijskim greškama.

.

7. Literatura

[1] RANSAC, 14.11.2008., *Wikipedia, the free encyclopedia*, http://en.wikipedia.org/wiki/RANSAC

[2] Fisher M., Bolles R., Random Sample Consensus: A Paradigm for Model Fitting with Apphcatlons to Image Analysis and Automated Cartiography, Communications of the ACM, Volume 24, Number 6 (1981), str. 381-395, http://www.ai.sri.com/pubs/files/836.pdf

[3] Charles V. Steward, Robust parameter estimation in computer vision, SIAM Review, Vol. 41, No 3 (1999), str. 513-537

[4] Scrhabel R., Wahl R., Klein R., Efficient RANSAC for Point-Cloud Shape Detection,The Eurographics Associoation and Blackwell Publishing , Volume 0 (1981), Number 0, str. 1-12, http://cg.cs.uni-bonn.de/aigaion2root//attachments/schnabel-2007-efficient.pdf

[5] Zuliani M., Kenney C., Manjunath B.S. i drugi, Using Level Sets and RANSAC to Estimate Affine Transforms Between Images, Research Department Naval Air Warfare Center, China Lake, CA

8. Sažetak

U ovom radu baviti ćemo se algoritmom RANSAC, koji se koristi za procjenu parametara nekog matematičkog modela iz skupa podataka. On je pogodan za interpretaciju skupa podataka u kojem postoji veća količina podataka sa velikim odstupanjem od traženog modela. Proučiti ćemo njegovu izvedbu, prednosti i nedostatke, te potencijalnu primjenu.