Lazar Marković

Implementacija preprocesivnog dela za rekonstrukciju 3D objekta na osnovu 2D izometrijske projekcije

Proces rekontrukcije 3D objekta na osnovu 2D izometrijske projekcije, tehničkog crteža, sastoji se od više koraka, a prvi i najvažniji je interpretacija, odnosno preprocesiranje crteža. Detektovanje lažnih temena na crtežu i prepoznavanje stranica objekta su dve operacije koje čine preprocesivnu jedinicu. Ovaj rad opisuje implementaciju i diskutuje rezultate algoritama koji obavljaju ove operacije. Da bi preprocesiranje tehničkog crteža bilo uspešno i da bi se dobili ispravni rezultati, važno je da crtež poštuje neka osnovna pravila. Rad preprocesivne jedinice se može poboljšati i implementiranjem algoritama za analizu podebljanja linija kao i za detektovanje nepravilnosti na crtežu.

Uvod

3D rekonstrukcija je važna grana oblasti računarskog vida. Kako većina ideja u inžinjerskom dizajnu krene crtežom ili skicom, one predstavljaju lako ostvarljiv vid prezentacije i evolucije same ideje. 3D rekonstrukcija ima za ulogu da oponaša ljudsko razumevanje dvodimenzionalnih reprezentacija trodimenzionalnih objekata. Cilj ovog procesa je da omogući brži i prirodniji način realizovanja i prenošenja ideja.

Proces 3D rekonstrukcije se sastoji iz više delova, od kojih je ključna interpretacija grafa koji predstavlja skicu. Taj deo se naziva preprocesivnim delom. Prema radu Lipsona (1991), postoji više različitih algoritama koji obavljaju 3d rekonstrukciju. Zajedničko za sve njih je to što je potrebana precizna, jasna i efikasna preprocesivna jedinica koja će isporučivati potrebne informacije rekonstruktoru. Preprocesivni proces sastoji se iz: ispravljanja ulaznih informacija, formiranja praktičnih struktura podataka i dela za pronalaženje trodimenzionalnih strana u grafu povezanosti temena. Kao polaznica za rešavanje poslednjeg problema biće upotrebljen algoritam iz literature (Varley i Company 2010), koji se pokazao optimalnim.

Strukture podataka

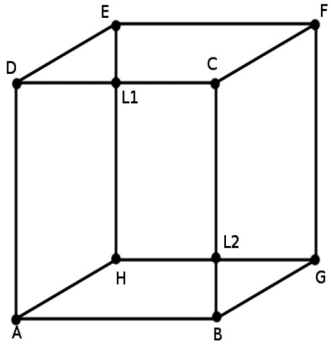
U prikupljačima podataka kao što je ova jedinica, a i uopšte, veoma je važno napraviti optimalne strukture podataka iz kojih se lako mogu dobijati podaci potrebni za sledeći korak nekog procesa ili u ovom slucaju za 3D rekonstrukcije. Korišćene strukture su:

- teme – sadrži dvodimenzionalne koordinate jednog temena i njegov redni broj;
- vektor – sadrži pravac i dužinu stranice;
- linija – sadrži koordinate početka i kraja linije;
- iskorišćenost svakog elementa temena – sadrži redni broj temena i informaciju o njegovoj iskorišćenosti;
- graf – sadrži niz temena, broj temena i kvadratnu matricu povezanosti temena;
- informacije o susedima – sadrži koordinate i redne brojeve temena;
- graf suseda – sadrži matricu tipa „informacije o susedima”. Redovi u ovoj matrici predstavljaju redne brojeve temena, a kolone redom temena sa kojima su one spojene. Graf suseda takođe sadrži niz koji sadrži broj suseda za svako teme;

Lazar Marković (1996), Valjevo, Jakova Nenadovića 48, učenik 2. razreda Valjevske gimnazije

– strana objekta – sadrži niz temena, informacije o iskorišćenosti i kompletnosti jedne strane;
– lista strana – sadrži niz strana, početnu dužinu niza i kontrolnu dužinu niza.

Što se tiče ispunjavanja ovih struktura, proces je krajnje jednostavan i može se postići učitavanjem iz datoteke ili transferom podataka iz neke druge klase, npr. klase iz navedenog rada (Lipson 1991), koja na slici skice otkiva temena i stranice, i predstavlja deo 3D rekonstrukcije kao oblasti.



Slika 1

Algoritmi

Algoritmi koji su opisani i provereni u ovom radu su: algoritam za uklanjanje laznih temena i algoritam za pronalazenje strana trodimenzionalnog objekta sa dvodimenzionalnih grafova tj. skica.

Algoritam za uklanjanje lažnih temena

Ovaj algoritam detektuje i brise lažna temena iz grafa. On se vrši nakon ispunjavanja grafova i pre ispunjavanja drugih struktura. Lažna temena su preseci linija nastali predstavljanjem trodimenzionalnog objekta u dvodimenzionalnoj ravni. Ovaj problem se najbolje vidi na sledećoj slici.

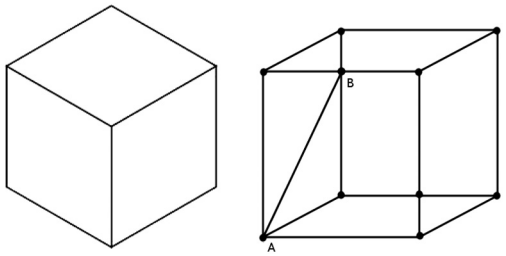
Na slici 1 je prikazana struktura početnih informacija iz strukture graf, vidimo da su temena L1 i L2 takođe uključena u graf povezanosti iako ona u realnosti ne postoje. Zato se nazivaju lažnim temenima. Algoritam se može predstaviti sledećim pseudokodom:

Vertikalnost se definiše računanjem kosinusa ugla koji zaklapa linija od jednog temena do drugog temena iz strukture graf ili strukture graf suseda. Prema Lipsonu (1991) prag paralelnosti dve stranice je razlika od 7° između vertikalnosti linija koje se porede. Dakle što je razlika manja od 7° to su linije paralelnije.

Problemi. Greške koje mogu nastati tokom izvršavanja ovog algoritma će se javiti ukoliko se ne ispoštuju sledeća pravila o ulaznim podacima:
– Ulazni podaci ne smeju sadržati viškove, odnosno linije koje nemaju geometrijsku svrhu u skici/grafu.
– Svako teme mora biti spojeno sa najmanje 2 druga.
– Dvodimenzionalni prikaz, tj. skica mora imati vidljive sve komponente. To znači da se moraju videti sva temena i stranice objekta na slici.

Na sledećim ilustracijama možemo videti primere na kojima ovaj algoritam neće funkcionisati, a samim tim ni 3D rekonstrukcija.

```
Formiraj graf suseda
  Za svako teme T u strukturi graf suseda
    Ukoliko je teme povezano sa parnim brojem suseda
      Za svako teme F takođe iz graf suseda koje je povezano sa T
        Izračunaj vertikalnost linije FT
          Za svako teme K koje je isto sused sa T
            Uporedi vertikalnost FT sa vertikalnosti KT ukoliko je razlika ovih vrednosti
            manja od 7°
              Ukloni teme T iz grafa i iz grafa suseda
              Spoj temena koja su bila spojena preko temena K u grafu i grafu suseda
              Obustavi poređenje za tekuće teme T.
```



Slika 2

Na slici 2 levo vidimo primer gde je prikazana kocka na kojoj prednja strana prekriva zadnju stranu, tj. gledamo u veliku dijagonalu kocke. Ovakav primer bi prouzrokovao pogrešno rešenje jer se ne dostavlja potpuna informacija o objektu, tj. skici. Primera radi, proces 3D rekonstrukcije bi ovu sliku prepoznao kao vrstu piramide, a ne kao kocku.

Na slici 2 desno 3 vidimo primer koji sadrži suvišnu liniju AB koja neće omogućiti prepoznavanje temena B kao lažnog temena, sto će prouzrokovati pogrešno rešenje.

Algoritam za pronalaženje strana 3D objekata u 2D grafovima

Ovaj algoritam treba da pomoću podataka iz izgenerisanih struktura pronađe sve strane koje bi objekat na skici imao u realnosti. Kako se uobičajeni algoritam za rešavanje navedenog problema – Dijkstrin algoritam (Varley 2009) pokazao lošijim u odnosu na noviji algoritam Petera A. C. Varley-a (Varley i Company 2010), u ovom radu diskutovan je i implementiran Varley-ev algoritam.

Uvod u algoritam. U ovom segmentu su definisani pojmovi i operacije koje će se koristiti u objašnjenju i implementaciji navedenog algoritma.

Poluivica podrazumeva strukturu strana objekta koja nije dovršena i sadrži najmanje jedan par spojenih temena, primera radi: ABC, AB i ABCD su poluivice, a ABCDA je *ciklus*, tj. zatvorena strana objekta i kako je on strana objekta on mora biti planaran. Poluivice imaju i pravac, tako da se uvodi restrikcija da se poluivica ne sme spojiti sa inverznom poluivicom, npr. AB sa BA, itd.

Da bi algoritam funkcionisao, pretpostavlja se da se od ulaznih podataka može načiniti najmanje jedan ciklus, tj. da ulaz čini najmanje tri temena koja su spojena. *Prioriteti svakog temena* računaju se na osnovu broja temena sa kojima je ono spojeno.

Operacija *nastavljanje* kombinuje poluivice A i B produžavajući A sa B. Ovaj proces se odvija na sledeći način:

- Početno teme od A′ je početno teme A (A′ je novo A).
- Krajnje teme od A′ je krajnje teme B.
- Temena između početka i kraja poluivice B se nižu redom na A.
- Dodeljuje se novi prioritet poluivice A′ koji bi prema radu Varley i Company (2010) trebalo da bude manji od ukupne vrednosti spojenih temena.
- Poluivica se markira kao iskorišćena.

Nastavljanje se vrši jedino ako je krajnje teme A isto kao početno teme B.

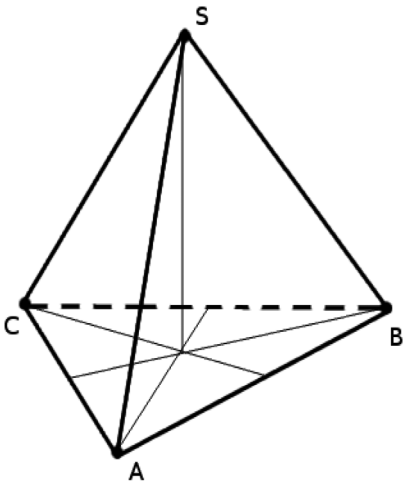
Operacija *zatvaranje* je zapravo nastavljanje, nakon koga se stvara ciklus i poluivica na kojoj je nastao ciklus se markira kao završena.

Algoritam. Pre predstavljanja samog algoritma definisaćemo i predstaviti podprocese glavnog algoritma:

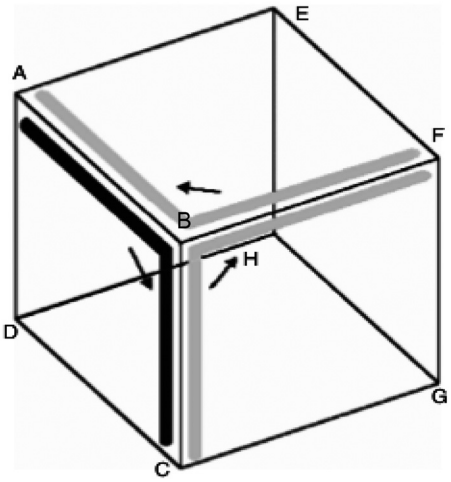
Inicijalizacija podrazumeva formiranje početne liste strana, koja se sastoji od svih ivica koje čine graf, odnosno svih spojeva među temenima u početnom grafu i njihovog inverza. Ovo je pojašnjeno sledećom slikom i primerom koje je prati. Na primeru za trostranu piramidu (slika 3) lista strana bi izgledala ovako: AB, BA, BC, CB, CA, AC, AS, SA, BS, SB, CS, SC.

Proces inicijalizacije se nastavlja pravljenjem početnog rešenja, odnosno prvog nastavljanja. U slučaju na slici početno rešenje, kako se AB i BA po restrikciji ne mogu spojiti, bi se dobilo spajanjem AB sa BC, dakle ABC.

Primorana nastavljanja su posledica nastavljanja ili zatvaranja neke strane objekta. Ovaj proces se vrši ukoliko je nastalo novo opkoljeno teme Ukoliko je nastalo novo opkoljeno teme u nekoj strani objekta u listi strana, vrši se nastavljanje svake dve poluivice koje sadrže to novo opkoljeno teme i koje se mogu nastaviti. Na primer B iz ABC (slika 4) je novonastalo opkoljeno teme, primorana nastavljanja za B su AB+BF i CB+BF).



Slika 3



Slika 4

Voluntarna zatvaranja su zatvaranja koja stvaraju ciklus od neke poluivice kojoj nedostaje samo jedno nastavljanje da postane ciklus.

Ispitivanje hipoteze prioriteta je proces u kome se prvo izabira poluivica sa najvećom primarnošću, onda se za nju se traži dvočlana poluivica sa najvećom vrednosti parenja i na kraju one se nastavljaju. Ispitivanje hipoteze prioriteta se koristi ukoliko nijedan drugi metod ne daje bilo kakvo rešenje i njegova primena će biti bolje objašnjena u glavnom algoritmu ispod.

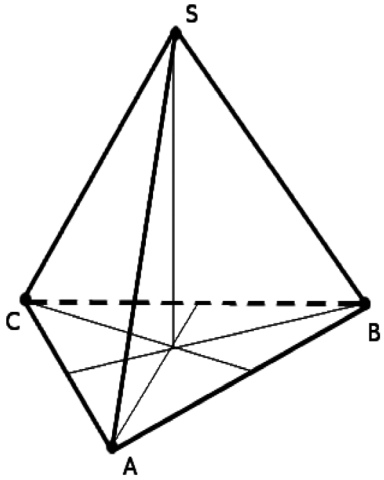
*Vrednost parenj*a predstavlja vrednost koja se dobija poređenjem paralelnosti neke dvočlane poluivice i nekog para temena iz višečlane ivice za koju se ispituje vrednost parenja.

Problemi sa kojima se ovaj algoritam može susresti opisali su detaljno Varlkey i Company (2010).

Primer rada. Za testiranje gore navedenog algoritma i boljeg pojašnjenja navedenih operacija koristiće se primer tetraedra (slika 5), dok je primer kocke testiran u radu Varley i Company (2010).

GLAVNI ALGORITAM:

Izvrši inicijalizaciju i dodeli prioritete poluivicoma
Sve dok u listi strana postoji još neiskorišćenih i nedovršenih poluivica
Pretraži listu tražeći primorana nastavljanja
Ukoliko je pronadjen takav slučaj, izvedi nastavljanje na njima
U suprotnom, pretraži listu ražeći voluntarna zatvaranja
Ako su pronađeni takavi slučajevi, izvedi nastavljanje na njima
U suprotnom ispitaj hipotezu prioriteta
Ponavljaj
Pretraži listu tražeći primorana nastavljanja
Ukoliko je pronađen takav slučaj, izvedi nastavljanje na njima
Ukoliko je ova operacija stvorila ciklus markiraj ga kao završen
Ako nema primoranih nastavljanja
Ako postoji voluntarnih spajanja, uradi ih i napravi cikluse
U suprotnom, izađi iz trenutne unutrašnje petlje



Slika 5

1. Algoritam počinje generisanjem liste strana: AB, BA, BC, CB, CA, AC, AS, SA, SB, BS, CS, SC.
2. Zatim se formira prvo moguće nastavljanje, u ovom slučaju ABC. Lista strana je: ABC, BA, CB, CA, AC, AS, SA, SB, BS, CS, SC.
3. Traže se i obavljaju primorana nastavljanja. Spajamo BA sa SB i CB sa BS. Lista strana je sada: ABC, CBS, CA, AC, AS, SA, SBA, CS, SC.
4. U sledećoj iteraciji nema primoranih nastavljanja ali zato postoje voluntarna zatvaranja: ABC i CA, CBS i SC, SBA i AS. Lista strana je: ABCA, CBSC, SBAS, AC, SA, CS.
5. Obavljanjem primoranih nastavljanja spajaju se SA i AC, lista sada izgleda ovako: ABCA, CBSC, SBAS, SAC, CS.
6. U ovoj iteraciji nema primoranih nastavljanja, tako ta tražimo i obavljamo voluntarna zatvaranja: SAC+CS. Lista strana na kraju izgleda ovako: ABCA, CBSC, SBAS, SACS, što i jeste traženi rezultat.

Zaključak

U ovom radu su implementirana dva algoritma koja čine pripremni modul za proces 3D rekonstrukcije objekata sa skice. Tradicionalni algoritmi, koji uključuju pristup problemu preko genetskog algoritma, pristup na osnovu Djikstrinog algoritma za pronalaženje najkraćeg puta padaju na kompleksnijim primerima koji se detaljnije mogu videti u originalnim radovima (Liu i Tang 2005; Varley 2009). Ovde je opisani algoritam obuhvata veći opseg primera za koje daje tačno rešenje nego za razliku od dva gore navedena algoritma.

U budućem radu, kako bi preprocesivna jedinica bila zaista potpuna, potrebno je pronaći adekvatan algoritam koji bi iz slika efikasno ekstraktovao graf, i analizirao debljine linija, podebljanja i nepravilnosti i koristio ih kao dodatne podatke za uspešnije pronalaženje stranica i precizniju 3D rekonstrukciju.

Literatura

Lipson H. 1991. *Computer Aided 3D Sketching for conceptual design*. Haifa: Israel Institute of Technology

Varley P. A. C., Company P. P. 2010. A new algorithm for finding faces in wireframes. *Computer-Aided Design*, **42** (4): 279.

Varley P. A. C. 2009. An implementation of Dijkstra's algorithm for finding faces in wireframes. *Technical Note Regeo*, 04. Dostupno na http://www.regeo.uji.es/publicaciones/regeo04.pdf

Liu J., Tang X. 2005. Evolutionary search for faces from line drawings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27** (6): 861.

Lazar Marković

Preprocessing Unit Implementation for 3D Object Reconstruction from 2D Isometric Projection Drawings

The reconstruction process of a 3D object based on its 2D isometric projection – technical drawing – is a multiple-step process with one of the steps being both the first and the most important one: interpretation/preprocessing of the drawing. The two operations implemented in the preprocessing unit are the detection of false vertices and the detection of object's sides.

This paper describes the algorithms developed and used in the implementation, and presents the obtained results. In order to have successful preprocessing, the algorithms developed need the drawings to follow some basic rules. The results of the preprocessing unit can be improved by implementing additional algorithms to detect and analyze different line thicknesses and drawing irregularities.