Pronalaženje centra najvećeg kruga koji može da stane u dati prost poligon

Opis i implementacija algoritma u okviru kursa
Geometrijski algoritmi
Matematički fakultet

Nikola Mićić, 1086/2022
nikolamicic065@gmail.com
Mentor: Jelena Marković

Novembar 2022.

Sažetak

U ovom radu će biti opisan algoritam za pronalaženje centra najvećeg kruga koji može da stane u dati prost poligon i način na koji sam ga implementirao. Algoritam je implementiran u okviru kursa Geometrijski algoritmi, na master studijama Matematičkog fakulteta.

Sadržaj

1 Opis problema

Tema algoritma čiji će opis i implementacija biti opisani je pronalaženje centra najvećeg kruga koji može da stane u dati prost poligon P.

Poligon ili mnogougao je zatvorena izlomljena linija. Segmenti izlomljene linije se nazivaju stranicama poligona, dok su krajevi stranica temena poligona.
Prost poligon je poligon koji nema samoprseke, tj. ako:

- iz svakog njegovog temena ishode samo dve stranice
- stranice nemaju zajedničkih tačaka (temena ne pripadaju stranicama)
- temena ne leže na stranicama

Poligon je konveksan ako duž koja spaja svake dve njegove tačke, pripada (sa svim svojim tačkama) tom mnogouglu.
U suprotnom je nekonveksan, odnosno konkavan i u ovom algoritmu su razmatrani prosti poligoni koji su nekonveksni (konkavni), ali algoritam će funkcionisati i za konveksne proste poligone.

Problem predstavlja pronalazak tačke koja pripada unutrašnjosti poligona P i koja predstavlja centar najvećeg kruga koji se može upisati u dati prost poligon P.

2 Opis algoritma

Ključ rešenja ovog problema je pronalazak tačke koja će predstavljati centar najvećeg kruga koji se može upisati u dati prost poligon. Za tu tačku mora važiti:

- nalazi se unutar poligona
- najudaljenija je od svih tačaka svih stranica (ivica) poligona.

Objašnjenje ovih uslova leži u tome da su sve tačke koje pripadaju kružnici kruga podjednako udaljene od centra kruga, odnosno tačke koju tražimo. Po definiciji, najveći krug će imati najveći poluprečnik i dodirivaće poligon u najmanje dve tačke, tako da tražena tačka će biti tačka koja je najudaljenija od svih tačaka svih stranica poligona.

Algoritam se može podeliti u narednih 5 koraka:

1. Definišemo pravougaoni region R koji se prostire od donje leve tačke (x_{min},y_{min}) do gornje desne tačke (x_{max},y_{max}). Pri čemu je x_{min} najmanja x koordinata poligona P, x_{max} najveća x koordinata poligona P, y_{min} najmanja y koordinata poligona P, y_{max} najveća y koordinata poligona P.
2. Zatim R podelimo na matricu tačaka, tako što ćemo visinu i širinu pravougaone regije R podeliti odgovarajućim brojem n i zatim napraviti kombinaciju x i y koordinata. U ovom algoritmu je dužina i visina deljena brojem 20, na osnovu čega dobijamo ukupno 21 tačku po dužini i 21 po širini, čijom kombinacijom dobijamo matricu tačaka.
3. U ovom koraku se izbacuju sve tačke koje ne pripadaju poligonu P.
4. Od preostalih tačaka koje su u okviru poligona se nalazi tačka koja je najudaljenija od svih tačaka svih stranica poligona.

5. Oko pronađene tačke se kreira nov pravougaoni region R sa manjim granicama, tako što se dužina i širina od R u svakoj iteraciji smanjuju za faktor kvadratnog korena dvojke. Zatim se ponavljaju koraci od 2. do 5. sve dok se ne postigne odgovarajuća preciznost.

3 Implementacija algoritma

Implementacija algoritma će biti predstavljena po koracima.

3.1 Prvi korak algoritma - pravougaoni region R

Na početku su definisana privatna polja xmin, xmax, ymin, ymax, i vrednosti xmax i ymax su inicijalizovane na 0, a xmin i ymin inicijalizovane na max float vrednost preko std::numericlimits<float>::max().

U prvom koraku se prolazi kroz sve tačke poligona i proverava se da li su te tačke manje od trenutnih minimalnih x i y koordinata ili su veće od trenutnih maksimalnih x i y koordinata, ukoliko jesu dolazi do dodeljivanja tih koordinata.

Na taj način smo definisali pravougaoni region R koji se prostire od donje leve tačke (x_{min},y_{min}) do gornje desne tačke (x_{max},y_{max}).

3.2 Drugi korak algoritma - deljenje R-a na matricu tačaka

R delimo na matricu tačaka tako što delimo visinu i širinu pravougaone regije R. Definišemo promenljivu divider i inicijalizujemo je na broj podela kojih ćemo izvršiti po visini i širini R-a.

Zatim definišemo promenljive koje predstavljaju vrednost razmaka između dve x i dve y koordinate za novu matricu tačaka. Te vrednosti dobijamo oduzimanjem maksimalne od minimalne x, odnosno y koordinate i delimo promenljivom divider.

Na početku je definisan privatni vektor tačaka QPointF za x i y koordinate regije R (rectXs, rectYs). Zatim se u te vektore dodaju nove x,y koordinate počevši od xmin,ymin i zatim dodaju iterativno nove vrednosti veće za izračunatu vrednost razmaka sve dok se ne dođe do xmax, odnosno ymax.

3.3 Treći korak algoritma - izbacivanje tačaka koje ne pripadaju poligonu P

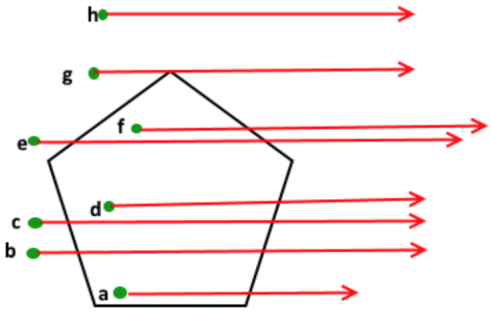
Nakon toga se kreira dupla petlja koja prolazi kroz sve dodate x, odnosno y koordinate iz vektora rectXs,rectYs i za svaku tačku se proverava da li pripada poligonu P ili ne. Ukoliko pripada tačke se dodaju u vektor polygonPoints, koje se kasnije koriste za traženje tačke koja predstavlja centar maksimalno upisanog kruga u poligon.

Algoritam korišćen za proveravanje da li tačka pripada unutrašnjosti poligona ili ne je vremenske složenosti O(n), gde je n broj temena datog poligona. Na slici 1 možete videti ilustraciju, a implementacija algoritma je sledeća [1]:

1. Za svaku tačku nacrtati horizontalnu liniju koja teži u beskonačnost
2. Izračunati broj presecanja te linije sa stranicama poligona

3. Tačka će biti unutar poligona ukoliko je broj preseka neparan ili ukoliko se tačka nalazi na poligonu. Ako nijedan od uslova nije tačak, tačka se nalazi van poligona.

Specijalni slučaj je tačka g kao na slici 1. Ukoliko ta linija preseca poligon u jednoj tački koja predstavlja teme tog poligona onda gledamo da li je ta tačka kolinearna sa stranicama za koje proveravamo, ukoliko jeste onda proveravamo da li se nalazi na toj liniji ili se nalazi van. U odnosu na to vraćamo true ili false.



Slika 1: Algoritam za proveravanje da li se tačka nalazi unutar poligona

3.4 Četvrti korak algoritma - traženje tačke koja je najudaljenija od svih tačaka svih stranica poligona

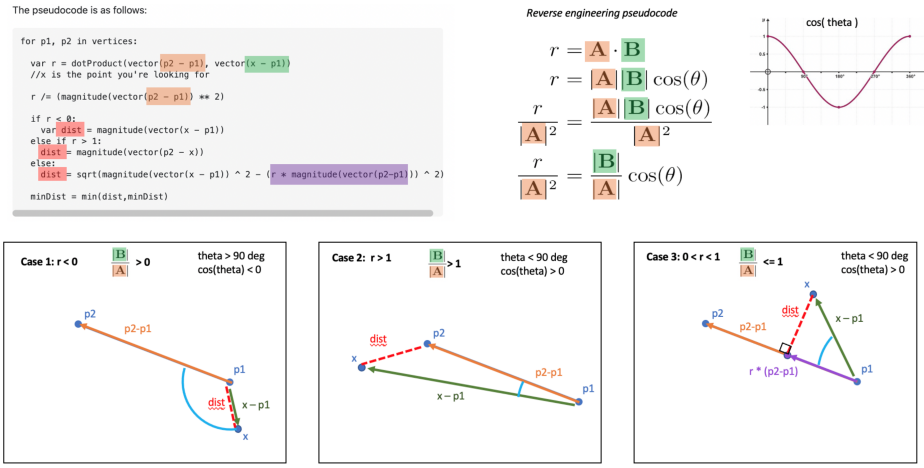
Ideja je bila izračunati udaljenost tačaka iz poligona od stranica poligona i odrediti onu koja ima najveću udaljenost u toj iteraciji. Tačke koje se proveravaju su tačke iz vektora polygonPoints.

Algoritam se svodi na matlabovu funkciju za računanje udaljenosti tačke od poligona [2]

Pseudokod i ilustraciono objašnjenje algoritma sam pronašao na jednom stackoverflow problemu i možete ga videti na slici 2.

Nakon što je izračunata distanca trenutne tačke od poligona, proveravamo da li je ta distanca veća od trenutne maksimalne i ako jeste, dodelimo promenljivoj koja čuva maksimalnu distancu tu novu vrednost.

Nakon što se provere sve tačke, imamo tačku sa maksimalnom udaljenošću od svih stranica, odnosno centar trenutnog najvećeg upisanog kruga u poligon. Nakon toga prelazimo na peti korak.



Slika 2: Algoritam za računanje distance tačke od stranica poligona

3.5 Peti korak algoritma - Ponavljanje koraka od drugog do petog dok se ne postigne odgovarajuća preciznost

Oko trenutne tačke koja ima najveću udaljenost do stranica poligona se kreira nov pravougaoni region R sa manjim granicama. Te granice se smanjuju tako što se u svakoj iteraciji dužina i širina smanjuju za faktor kvadratnog korena. Koraci od 2. do 5. se ponavljaju sve dok se ne postigne odgovarajuća preciznost.

U ovom algoritmu je preciznost definisana razlikom poluprečnika najvećeg kruga u dve susedne iteracije, i iteracije se zaustavljaju u momentu kada je razlika između prethodnog i trenutnog poluprečnika manja od 0.01.

4 Zaključak i rezultati

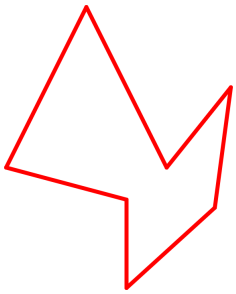
Ovim radom je predstavljen algoritam i implementacija za pronalaženje centra najvećeg kruga koji može da stane u dati prost poligon.

Na daljim slikama možete videti prikaz rada i rezultat algoritma za dati prost poligon.

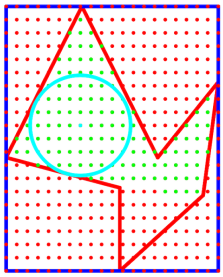
Učitan poligon izgleda kao na sledećoj slici 3.

Tokom izvršavanja algoritma matrica tačaka i pronađen trenutni centar kruga izgledaju kao na sledećoj slici 4.

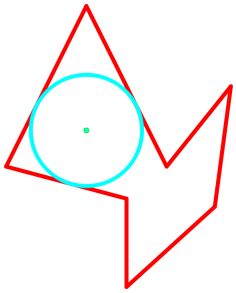
Nakon završenog algoritma dobili smo centar maksimalno upisanog kruga u dati poligon sa preciznošću 0.01 5.



Slika 3: Učitan poligon



Slika 4: Tokom izvršavanja algoritma



Slika 5: Završen algoritam - pronađen centar maksimalno upisanog kruga

Literatura

[1] Check if a given point lies inside a polygon. https://www.geeksforgeeks.org/how-to-check-if-a-given-point-lies-inside-a-polygon/.

[2] Point distance from polygon. https://www.mathworks.com/matlabcentral/fileexchange/19398-distance-from-a-point-to-polygon.