Collision Detection

Projekat na kursu Geometrijski algoritmi

Marija Katić

Matematički fakultet
Univerzitet u Beogradu

January 2021

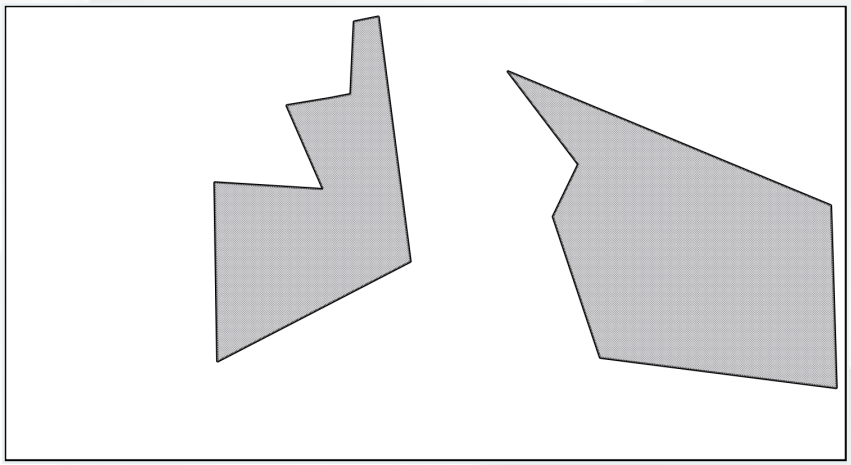
Sadržaj

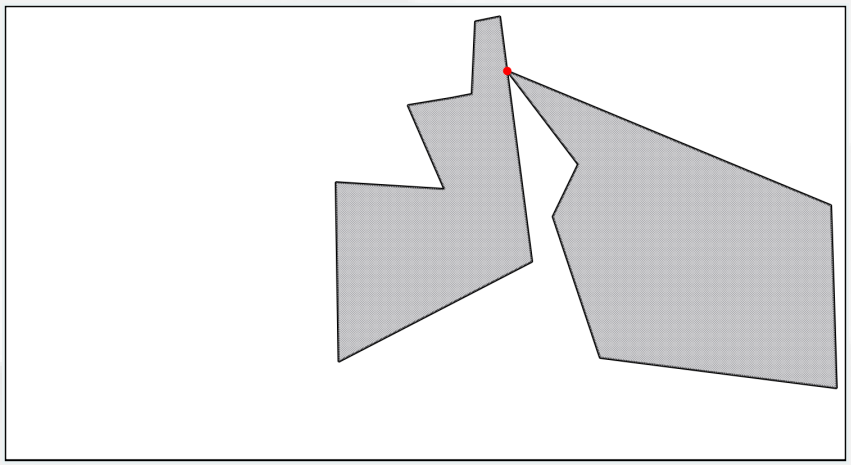
Opis problema

Data su dva prosta poligona P i Q koji se ne seku, i P se u celosti nalazi levo od Q. Algoritam, za takve poligone P i Q računa da li će se sudariti, ako se P kreće u pozitivnom smeru x-ose, i ako hoće, nakon koje udaljenosti i u kojoj tački će se desiti sudar.

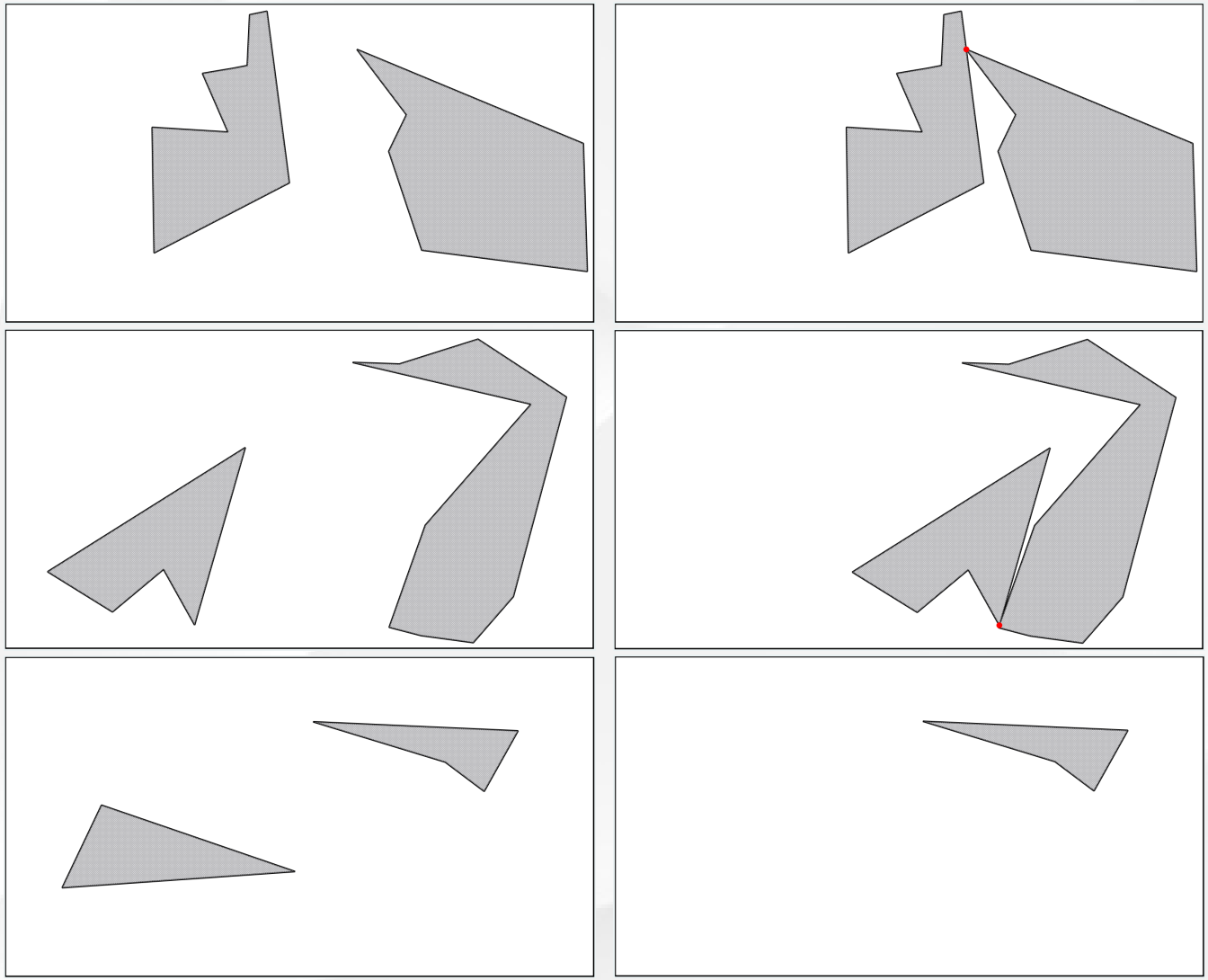
Ulaz algoritma: Dva prosta poligona koji se ne seku, zadati svojim temenima kojih ukupno ima N.

Izlaz algoritma: Udaljenost koju će levi poligon preći dok se ne sudare i jedna proizoljna tačka sudara.



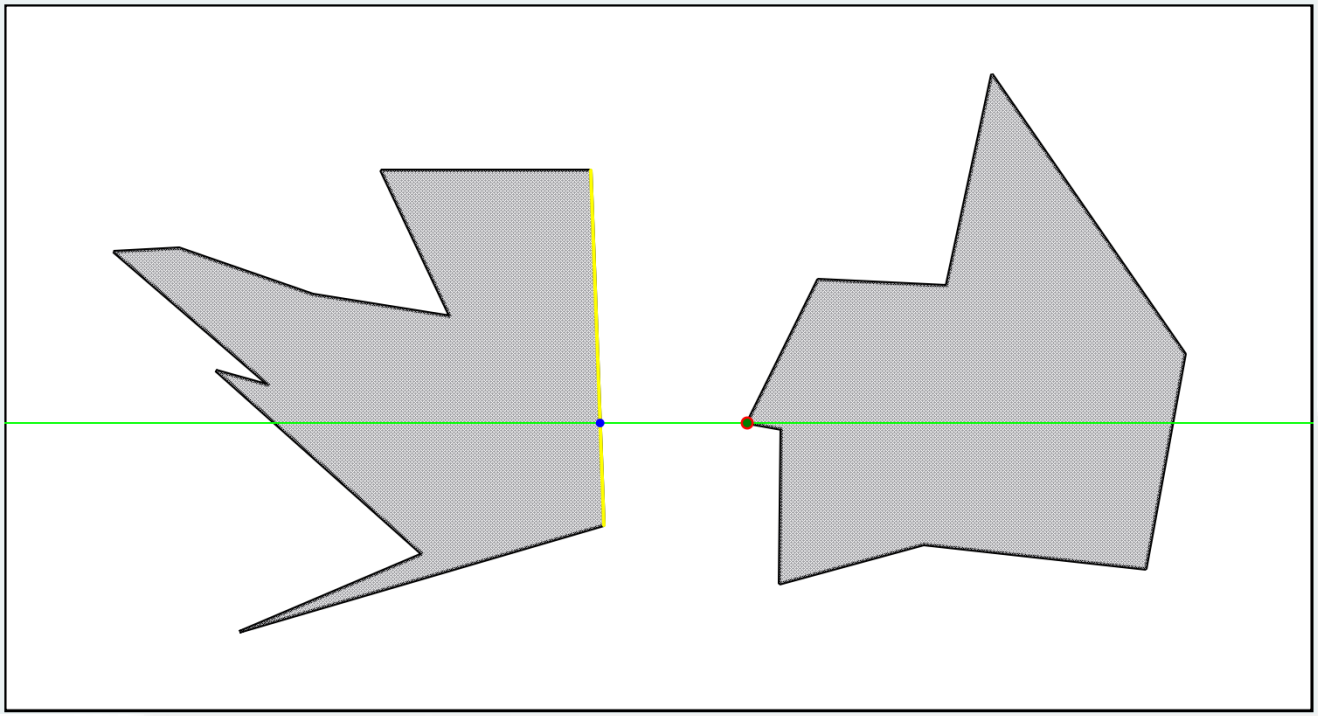


Opis problema



Naivna implementacija

Pronalaženje jedne tačke na izlomljenoj liniji (granici poligona) je neprekidan problem, pa ni rešenje grubom silom ne dolazi pravolinijski na pamet.
Ključni uvid je da će u koliziji kao dodirna tačka sigurno učestvovati makar jedno teme levog ili desnog poligona, pa problem svodimo na pronalaženje tog temena.



Naivna implementacija

Poligon P se kreće duž pravca x-ose, što znači da se tako kreće svaka njegova tačka. Dakle da bismo za neko teme iz levog poligona pronašli prvu tačku desnog poligona u koju ono udara ako se kreće udesno, treba da tražimo presek horizontalne prave kroz to teme sa desnim poligonom. Kako je kretanje relativan pojam (za posmatrača na levom poligonu se desni poligon kreće suprotno od smera x-ose), tako je pronalaženje prve tačke dodira za teme iz desnog poligona potpuno simetričan problem. Ono teme (od svih temena zajedno) koje je najmanje udaljeno od suprotnog poligona je teme koje će sigurno učestvovati u koliziji čitavih poligona.
Pps. da nije tako. To znači da se kolizija dešava ranije, a kako sve tačke poligona za isto vreme prelaze istu dužinu puta, znači da postoji tačka koja je bliža suprotnom poligonu, što je kontradikcija!

Naivna implementacija

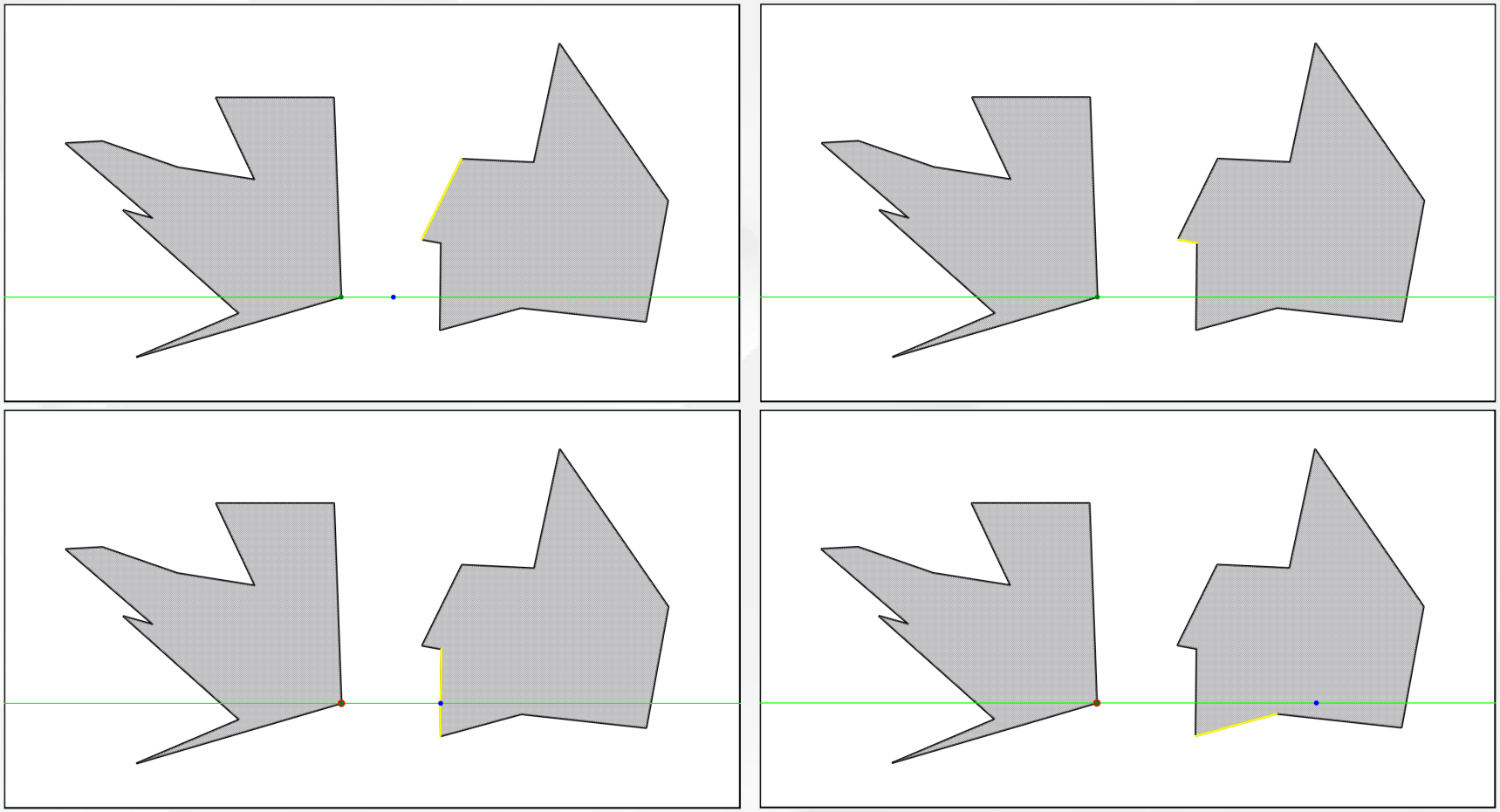
Kako pronalazimo koliko je teme udaljeno od suprotnog poligona? Najbliža tačka suprotnog poligona je na granici poligona (ne može biti u unutrašnjosti), pa je dovoljno pronaći ivicu poligona (uključujući i krajeve ivice) koja sadrži najbližu tačku. Dakle, dovoljno je proći kroz sve ivice suprotnog poligona, proveriti da li ih horizontalna prava kroz teme seče i ako da, koliko je presek udaljen od temena, i sačuvati koji je od svih preseka najbliži.
Primetimo da je zbog uključenja krajeva ivice obrađen i slučaj sudara "teme u teme". A primetimo da je obrađen i slučaj "ivica u ivicu" jer u tom preseku mora učestvovati makar jedna krajnja tačka.

Naivna implementacija

Naredni kod se pokreće za levi i za desni poligon simetrično, a ista minUdaljenost se ažurira za oba poligona.

```
Za svako teme v iz poligona P:
   Za svaku ivicu e iz poligona Q:
      Ako horizontalna prava kroz v seče ivicu e:
         d = udaljenost v od preseka
         Ako je d < minUdaljenost:
            minUdaljenost = d
```

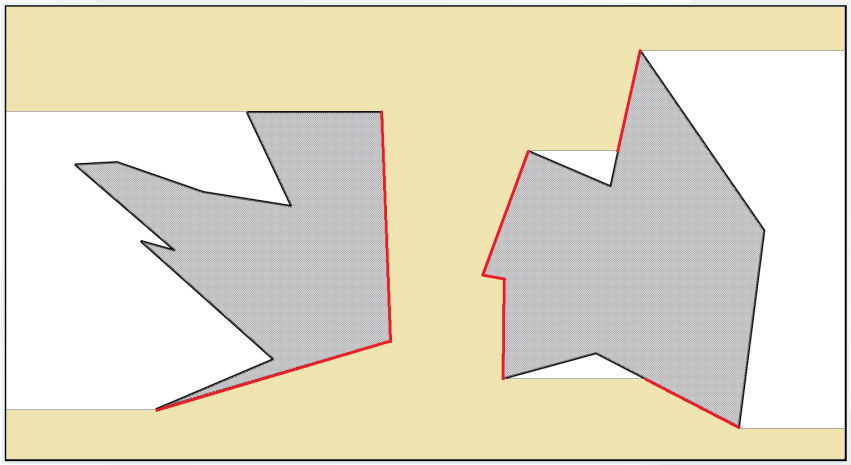
Naivna implementacija

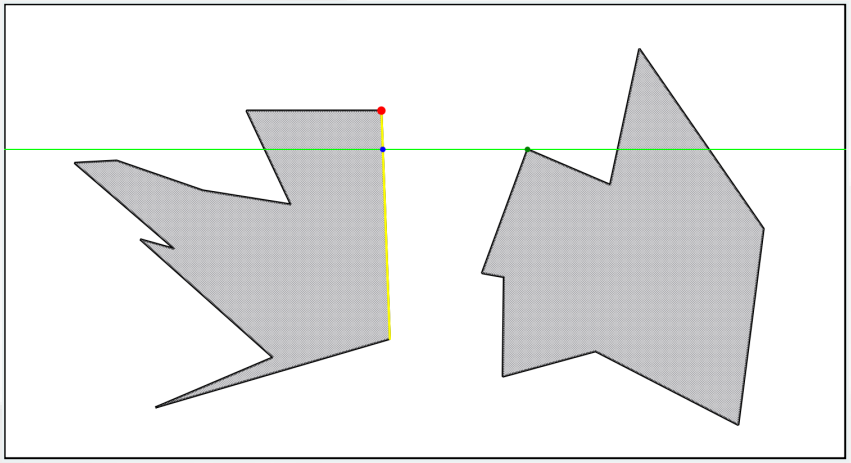


Efikasna implementacija

Prva ideja Izračunati "izloženu granicu" oba poligona (crveno na slici). Nakon toga se u vremenu O(N) može pronaći teme sudara. Ideja je bila izloženu granicu graditi iterativno, prolazeći kroz ivice poligona redom od najvišeg temena u smeru kazaljke na satu za levi i suprotnom za desni poligon. Ispostavilo se komplikovano.

Druga ideja Sweep line





Efikasna implementacija - Sweep line

Tačke događaja - Temena poligona P i Q

Status - Ivice koje brišuća prava trenutno preseca sortirane po x koordinati. Odvojene u dva stabla prema tome kojem poligonu pripadaju. Levo stablo sortirano opadajuće, desno rastuće.

U svakoj tački događaja računamo njegovu udaljenost do "najmanje" ivice iz suprotnog statusa i eventualno ažuriramo minUdaljenost. I ažuriramo odgovarajući status. Teme može biti:

"vrh" (da mu obe ivice idu na dole) Tada obe ivice dodajemo u status

"sredina" (jedna ivica iznad horizontale kroz teme a druga ispod) Tada gornju izbacujemo a donju dodajemo

"dno" (obe ivice iznad) Tada obe ivice izbacujemo iz statusa

Efikasna implementacija - Sweep line

Sortiranost ivica u statusu može biti ažurirana samo u ovim tačkama događaja jer se ivice ne seku (poligoni su prosti).
Ostali argumenti vezani za razmatranje udaljenosti SVIH temena ostaju isti kao u naivnoj implementaciji.
Naredni pseudo kod opisuje ceo algoritam.

```
Sortirati sva temena opadajuće po y koordinati
Za svako teme v (obilaziti opadajuće):
   ivica e = najbliža ivica iz suprotnog poligona (uzeti je iz statusa)
   d = udaljenost v od e
   Ako je d < minUdaljenost:
      minUdaljenost = d
   Ako je teme v "vrh":
      dodati obe ivice u status
   Ako je teme v "sredina":
      dodati donju ivicu
      izbaciti gornju ivicu
   Ako je teme v "dno":
      izbaciti obe ivice iz statusa
```

Detalji implementacije

- Poligoni se čuvaju kao nizovi temena, kao objekti klase `QPolygon`.
- Događaj je struktura podataka koja čuva pokazivač na teme, pokazivače na obe ivice iz temena, podatak o tipu temena (`enum struct { TOP, MIDDLE, BOTTOM };`), i podatak da li je teme u levom ili desnom poligonu (`enum struct { LEFT, RIGHT };`).
- Događaji se čuvaju u balansiranom stablu, u strukturi podataka `std::set`, komparator ih sortira prema y koordinati.
- Status je balansirano stablo (`std::set`) pokazivača na ivice poligona. Komparator ih sortira prema x koordinati preseka sa brišućom pravom (Brišuća prava je morala biti malo translirana na dole za poređenje)
- Moraju da postoje dva statusa, jedan za ivice levog poligona, drugi desnog.

Analiza složenosti algoritama

Naivni algoritam: Vremenska složenost algoritma je $O(N^2)$.

Prolazi se kroz sva temena kojih ima N, i za svako teme se prolazi kroz oko N/2 ivica.

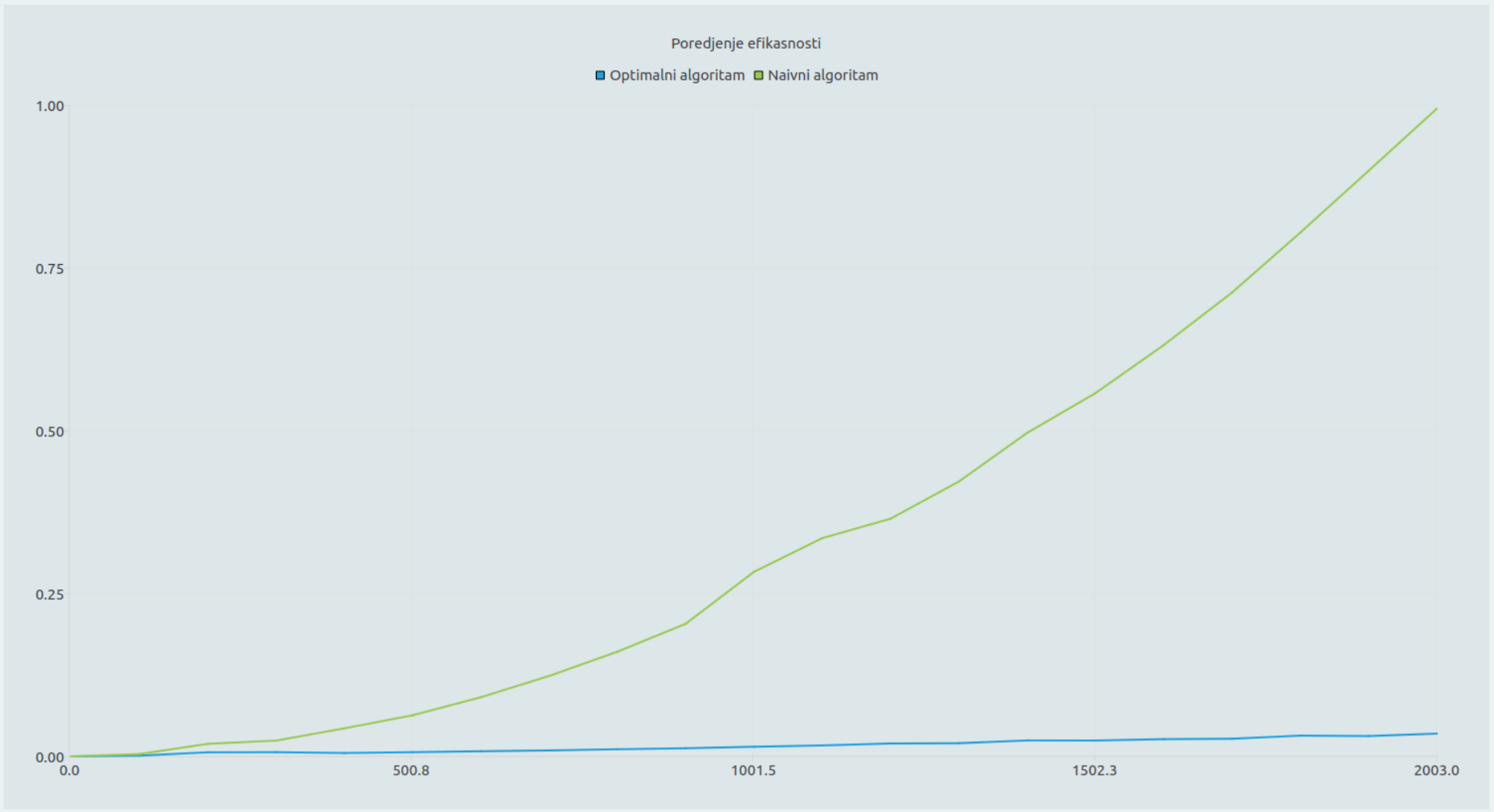
Memorijska složenost je $O(n)$.

Efikasni algoritam: Vremenska složenost algoritma je $O(NlogN)$.

Ubacivanje temena u balansirano stablo je ukupno složenosti $O(NlogN)$. Za svako teme se izvodi konačno mnogo operacija vremenske složenosti $O(logN)$ - izbacivanje i ubacivanje ivica u balansirano stablo (status).

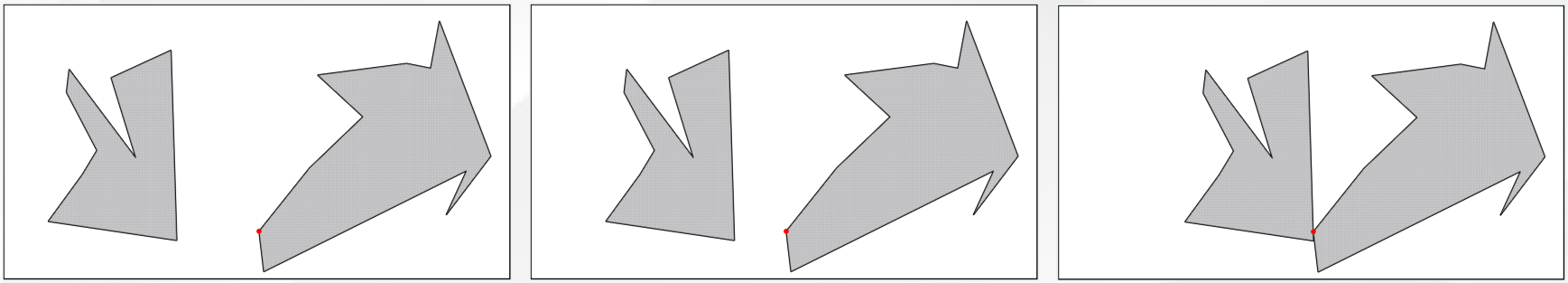
Memorijska složenost je $O(n)$.

Analiza složenosti algoritama



Vizuelizacija algoritama

Pri svakom razmatranju novog temena se ažurira slika. Ističe se trenutno razmatrano teme (tamno zeleno), crta se horizontalna prava kroz teme, ističe se teme koje trenutno ima najmanju udaljenost od suprotnog poligona (crveno), ističe se ivica sa kojom se razmatra presek (žuto).
Na kraju algoritma, da bi se ilustrovala tačnost, levi poligon se translira za nađenu udaljenost i ilustruje sudar.



Pokretanje programa i izlaz

Algoritam se pokreće iz grafičkog korisničkog interfejsa. Može se koristiti sa unapred zadatim brojem N (ukupnim brojem temena) ili podrazumevanim brojem $N = 20$. Može se odabrati pseudoslučajan odabir polinoma, a mogu se polinomi i opisati u ulaznoj datoteci.
Ulazne datoteke moraju biti sledećeg oblika: sadrže dva reda celih brojeva razdvojenih razmacima. Brojevi su parovi koordinata temena, i temena su poređana po redosledu obilaženja granice poligona, nebitno u kom smeru i od koje početne tačke.
Izlaz iz programa je jedan realan broj koji predstavlja udaljenost od trenutka kolizije i proizvoljna tačka koja će učestvovati u koliziji. Smešteni su u privatnim metodama klase koja predstavlja ceo algoritam, i može im se pristupiti pomoću get metoda.

Testiranje

Unit testovi su realizovani pomoću Google Test Framework-a. Svi testovi upoređuju rezultat naivnog i optimalnog algoritma - udaljenost za koju treba translirati levi poligon do sudara. Prošlo je nekoliko hiljada testova sa pseudoslučajno generisanim poligonima i nekoliko testova sa "ručno" definisanim ulazima.
Pomoću testova su otkrivene tri greške. Najzanimljivija se odnosi na slučaj horizontalnih ivica u poligonu. Metod `intersects` klase `QLineF`, kada su duži kolinearne vraća da su duži paralelne i ne menja tačku preseka koja se šalje kao argument, tako da se greška uspešno potkrala do trenutka bavljenja tim specijalnim slučajem.
Za potrebe testiranja napravljeno je nekoliko ulaznih datoteka. Pokriveni su specijalni slučajevi praznog poligona, poligona koji se ne sudaraju, poligona od dva i jednog temena, poligona sa horizontalnim ivicama, poligona koji se sudaraju "teme u teme".