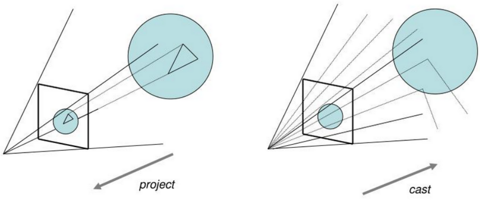
Računarska grafika
Prostorne strukture podataka

Vesna Marinković

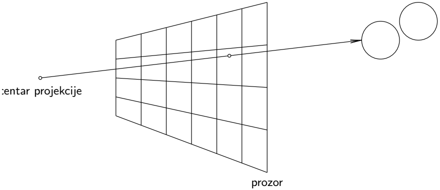
Podsetimo se...

- Postoje dve tehnike za prikazivanje objekata na ekranu
- Rasterizacija
  - elementi scene se projektuju ka kameri
  - za svaku primitivu (npr. trougao) utvrditi koje piksele treba osvetliti
- Tehnika praćenja zraka (rej trejsing)
  - zraci se iz kamere puštaju ka sceni
  - za svaki piksel utvrditi koje primitive se vide



Rej kasting algoritam

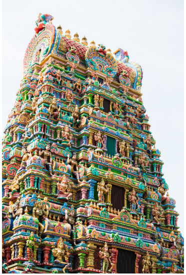
- Rej kasting algoritam utvrđuje da li je neka površ vidljiva praćenjem imaginarnog zraka svetlosti od otvora kamere do objekta na sceni
- Za svaki piksel prati se po jedan zrak od centra projekcije kroz sam piksel do najbližeg objekta na sceni
- Boja piksela se postavlja na boju najbližeg objekta
- Pun rekurzivni algoritam koji obrađuje i refleksiju i prelamanje svetlosti i senke naziva se rej trejsing



Rej kasting algoritam

- Rej kasting algoritam predstavlja direktan odgovor na upit preseka kojim se traži prvi presek zraka i primitive sa scene
- Naivni algoritam: ispitujemo presek zraka sa svakom od N primitiva (trouglova) i održavamo najbliži presek
- Složenost: $O(N)$ za svaki zrak
- Nepraktično za iole složenije scene

Upit preseka za složene scene



- U realnosti, scene mogu biti jako složene, sa puno detalja
- Kako se nositi sa ovolikim nivoom detaljnosti iz ugla vremenske složenosti?
- Da li možemo postići bolju vremensku složenost?

Mala digresija

- Dat je skup S od N celih brojeva i broj k. Odrediti element niza S koji je po vrednosti najbliži broju k

Mala digresija

- Dat je skup S od N celih brojeva i broj k. Odrediti element niza S koji je po vrednosti najbliži broju k
- Naivni algoritam: iteriramo kroz elemente niza S, računamo razliku između tekućeg elementa niza S i k, i održavamo najmanju razliku
- Složenost: $O(N)$
- Da li možemo bolje?

Mala digresija

- Dat je skup S od N celih brojeva i broj k. Odrediti element niza S koji je po vrednosti najbliži broju k
- Naivni algoritam: iteriramo kroz elemente niza S, računamo razliku između tekućeg elementa niza S i k, i održavamo najmanju razliku
- Složenost: $O(N)$
- Da li možemo bolje?
- Sortiramo niz S, pa radimo binarnu pretragu – složenost $O(\log N)$

Mala digresija

- Dat je skup S od N celih brojeva i broj k. Odrediti element niza S koji je po vrednosti najbliži broju k
- Naivni algoritam: iteriramo kroz elemente niza S, računamo razliku između tekućeg elementa niza S i k, i održavamo najmanju razliku
- Složenost: $O(N)$
- Da li možemo bolje?
- Sortiramo niz S, pa radimo binarnu pretragu – složenost $O(\log N)$
- Primetimo: sortiranje niza od N elemenata je složenosti $O(N \log N)$
- Međutim, sortiranje se radi jednom, a ako je potrebno rešiti $O(N)$ upita preseka, amortizovana složenost sortiranja je $O(\log N)$
- Da li primitive na sceni možemo reorganizovati tako da se upit preseka realizuje efikasnije?

Motivacija

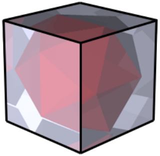
- Skupe operacije (rej kasting, upit preseka)
  - Složene scene (milioni objekata)
  - Veliki broj operacija (stotine miliona u sekundi)
- Smanjenje složenosti preprocesiranjem podataka
  - Prostorne strukture podataka: strukturirati objekte u prostoru
  - Što ranija eliminacija kandidata
  - Smanjenje složenosti na $O(\log n)$ u proseku
  - Složenost u najgorem slučaju i dalje $O(n)$

Prostorne strukture podataka

- Prostorne strukture podataka predstavljaju višedimenziono uopštenje klasičnih uređenih struktura podataka
- Cilj im je smanjenje vremena izvršavanja upita po cenu povećanja prostora, pa se nazivaju i prostorne ubrzavajuće strukture podataka
- Ne postoji najbolja prostorna struktura podataka – različite prostorne strukture podataka su pogodne za različite upite i različite raspodele primitiva na sceni
- Njihova primarna primena je za potrebe renderovanja i animacije u računarskoj grafici; međutim danas se koriste i u drugim oblastima

Granični opseg

- Ideja graničnih opsega je umetnuti složene objekte u jednostavnije
- Računamo npr. najmanju kocku koja sadrži date primitive
- Ukoliko zrak ne seče granični opseg, sigurno ne seče nijedan objekat u njegovoj unutrašnjosti
- Ukoliko zrak seče granični opseg, postoji mogućnost da seče neku primitivu
- Efikasnost zavisi od verovatnoće da zrak pogodi granični opseg, a ne i sadržane objekte (manji broj lažno pozitivnih rezultata)
- U najgorem slučaju složenost upita preseka ostaje $O(N)$

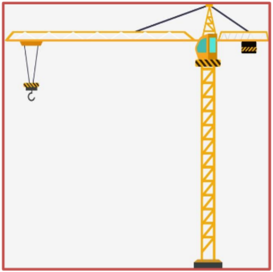


AABB

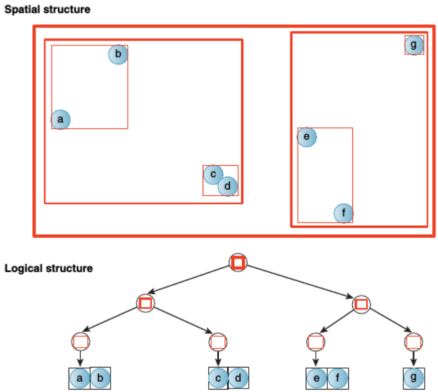
- Posebno je jednostavno odrediti granične opsege poravnate sa koordinatnim osama: AABB (Axis Aligned Bounding Boxes)

AABB

- Posebno je jednostavno odrediti granične opsege poravnate sa koordinatnim osama: AABB (Axis Aligned Bounding Boxes)
- Određujemo minimum i maksimum x, y i z koordinata svih temena – čuvamo samo 6 brojeva u pokretnom zarezu
- Konstrukcija graničnog opsega je takođe složenosti $O(N)$, ali se on računa jednom, a možemo imati veliki broj upita preseka
- Pogodni su za objekte koji imaju tesne granične opsege

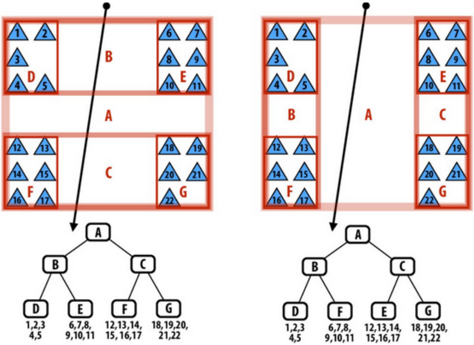


Hijerarhija graničnih opsega



- Cilj je hijerarhijski primeniti prethodnu strategiju
- Hijerarhija graničnih opsega je prostorno stablo nastalo rekurzivnim ugnježdavanjem graničnih opsega

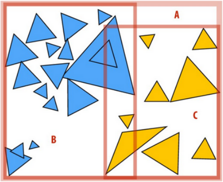
Hijerarhija graničnih opsega



- Listovi sadrže mali broj primitiva
- Unutrašnji čvorovi čuvaju granični opseg svih primitiva u podstablu
- Primitive se mogu grupisati na različite načine
- Koja od dve prikazane strukture ima bolja svojstva?

Hijerarhija graničnih opsega – konstrukcija

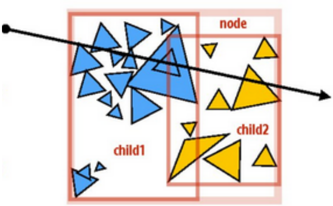
- Pristup odozdo naviše – od listova ka korenu, grupisanjem graničnih opsega susednih objekata sve dok se ne ograniči kompletna scena
- Pristup odozgo naniže – rekurzivnom podelom skupa primitiva na dva podskupa korišćenjem neke heuristike
- Iako se primitive dele u disjunktne podskupove, granični opsezi se mogu preklapati u prostoru



- Pogodna je za dinamičke scene jer se granični opsezi lako ažuriraju

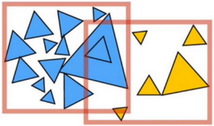
Hijerarhija graničnih opsega – upit preseka

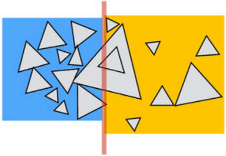
- Ako je čvor list, onda za svaku primitivu iz lista proveravamo da li je zrak seče
- Ako zrak ne seče tekući granični opseg ili je dalji od trenutno najbližeg preseka, ne nastavljamo rekurzivno pretragu u sadržanim opsezima
- Inače, rekurzivno nastavljamo sa sadržanim opsezima i primitivima
- Decu obrađujemo u redosledu kojim je zrak sekao granične opsege jer je tu veća verovatnoća da nađemo prvi presek
- Nekad moramo odrediti presek i sa drugim graničnim opsegom jer presek može biti bliži



Dve vrste particionisanja

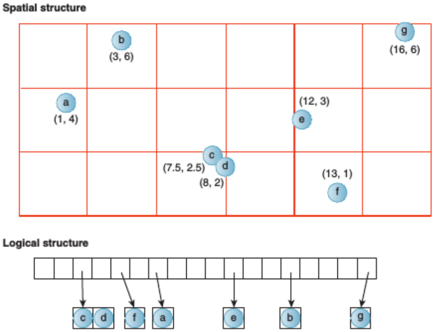
- Particionisanje primitiva u disjunktne podskupove
  - particije se mogu preklapati u prostoru
  - npr. hijerarhija graničnih opsega
- Particionisanje prostora u disjunktne regione
  - primitive mogu biti sadržane u većem broju regiona u prostoru
  - npr. uniformna mreža, *k*d stablo
  - prvi određeni presek je zaista i prvi presek zraka i primitive





Uniformna mreža

- Prostor možemo particionisati na ćelije jednake veličine i na taj način dobijamo uniformnu mrežu, odnosno rešetku (grid)
- Ove ćelije se nazivaju i vokseli
- Nije hijerarhijska struktura podataka

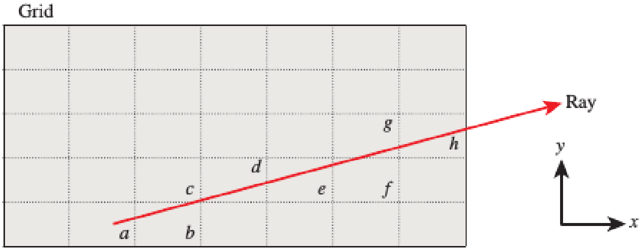


Uniformna mreža – konstrukcija

- Jednostavno se konstruiše i jednostavno se za datu tačku utvrđuje kojoj ćeliji mreže pripada
- Međutim, prostor potreban za skladištenje ove strukture je proporcionalan sumi broja primitiva i zapremine mreže
- Pogodna je za dinamičke podatke jer operacije dodavanja novih primitiva i brisanja nisu zahtevne ni vremenski ni prostorno

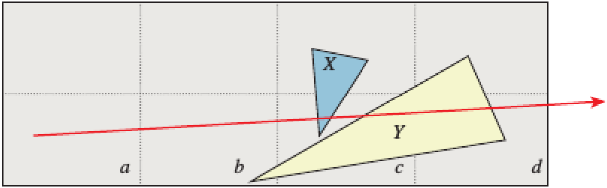
Uniformna mreža – upit preseka

- Da bi se odredio prvi presek zraka sa primitivom, ćelije mreže se moraju obilaziti u poretku u kom zrak ulazi u njih



Uniformna mreža – upit preseka

- Primitive se mogu protezati kroz više od jedne ćelije mreže
- Važno je u svakoj ćeliji testirati samo preseke koji se javljaju u toj ćeliji



Složenost upita preseka kod mreža

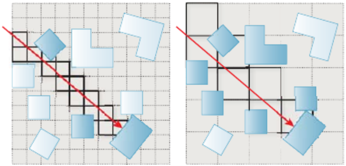
- Složenost upita preseka proporcionalna je broju ćelija mreže kroz koje zrak prolazi
- Cena svake iteracije je konstantna, te je algoritam od koristi kad ne očekujemo da zrak putuje predugo pre nego što "udari" u nešto
- Mreža sa g podela duž k dimenzija sadrži $O(g^k)$ ćelija
- Najduži put zraka je dužine $O(g)$
- U najgorem slučaju, za mrežu koja sadrži N primitiva, sve primitive se protežu kroz sve ćelije mreže duž dijagonale, a zrak ne preseca nijednu od njih te je cena upita preseka $O(g \cdot N)$
- Pogodna za scenu koja sadrži približno ravnomernu raspodelu primitiva po sceni koje teže tome da stanu u ćeliju mreže

Odabir rezolucije mreže

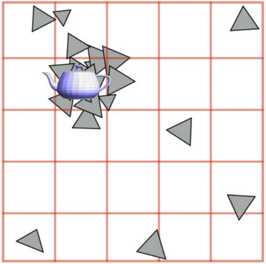
- Upit preseka je linearne složenosti u funkciji dužine zraka
- Upit preseka je linearne složenosti u funkciji broja primitiva u mreži
- Dva cilja:
  - minimizovati broj testiranja preseka sa mrežom
  - minimizovati broj testiranja preseka sa primitivama
- Ako je scena retka, veliki broj ćelija ne sadrži nijednu primitivu pa ako su ćelije male dimenzije puno vremena se gubi na njihovu proveru
- Ako je scena gusta, a ćelije su velike dimenzije, onda ćelije sadrže veliki broj primitiva
- Na osnovu same scene treba odabrati pogodnu rezoluciju

Odabir rezolucije mreže

- Ako je g veliko
    - kvadrati mreže su mali
    - smanjuje se broj primitiva koje treba testirati u svakoj nepraznoj ćeliji
    - povećava se broj ćelija koje treba ispitati
    - pogodnije za guste scene
- Ako je g malo
    - kvadrati mreže su veliki
    - zrak se brzo kreće kroz velike prazne prostore
    - povećava se broj primitiva u svakoj nepraznoj ćeliji koju treba testirati
    - pogodnije za retke scene



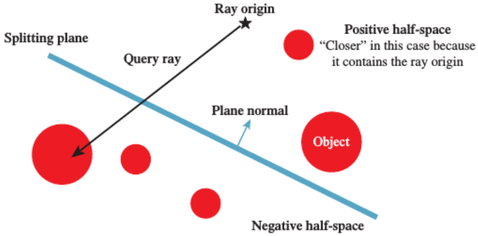
Neprilagodljivost uniformne mreže neuniformnoj raspodeli primitiva na sceni



- Particije uniformne mreže ne zavise od geometrije scene
- Problem: "Teapot in a stadium"
- Scena zauzima veliki prostorni opseg, a sadrži objekat visoke rezolucije koji zauzima jako mali deo scene (završi u jednoj ćeliji scene)

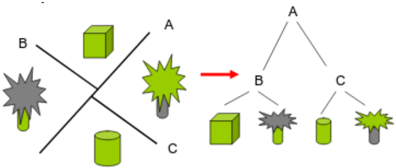
Binarno stablo prostornog particionisanja

- Binarno stablo prostornog particionisanja (BSP stablo) je prostorna struktura podataka koja se koristi za uređivanje geometrijskih primitiva na osnovu njihovog položaja i dometa
- Svaki unutrašnji čvor odgovara jednoj ravni razdvajanja (koja nije deo geometrije scene), a list geometrijskoj primitivi na sceni
- Ravan razdvajanja deli prostor na dva poluprostora

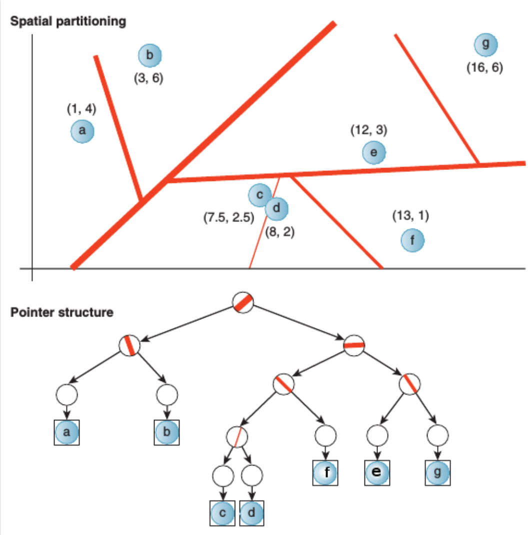


Binarno stablo prostornog particionisanja

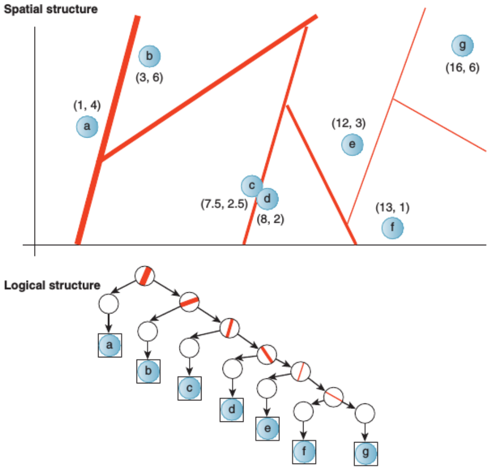
- Oba poluprostora biće dalje podeljena dodatnim ravnima razdvajanja
- Cilj je primitive što više razdvojiti od ostalih
- Unutrašnji čvor BSP stabla ima dva deteta
- Ako primitiva sa scene preseca ravan razdvajanja, onda je algoritam deli na dve primitive u odnosu na tu ravan



Primer binarnog stabla prostornog particionisanja u 2D



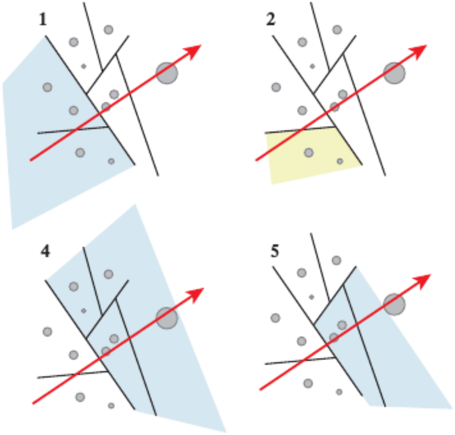
Primer neuravnoteženog binarnog stabla prostornog particionisanja u 2D



BSP stablo – konstrukcija

- Beskonačno mnogo mogućnosti za odabir ravni razdvajanja
- Izbor ravni razdvajanja zavisi od raznih faktora:
  - vrsta upita
  - raspodela podataka
  - da li optimizujemo najgori, najbolji ili prosečni slučaj
- Različiti ciljevi:
  - minimizovati vreme izvršavanja upita
    (ako se stablo gradi jednom, na početku)
  - minimizovati ukupno vreme izgradnje stabla i izvršavanja upita
    (ako se stablo gradi tokom rada programa)

BSP stablo – upit preseka



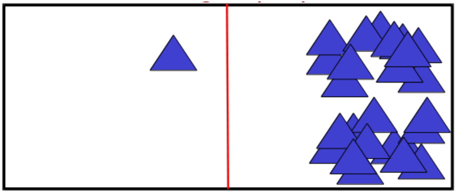
- Prosečna složenost: $O(\log n)$ po zraku

*k*d stabla

- Uvodimo ograničenja na izgled potprostora kod BSP stabla
- *k*d stablo (*k*d-tree) je BSP stablo kod koga su svi potprostori poravnati sa koordinatnim osama
- Bentley, 1970.
- Umesto 2d, 3d-stabla, zovu se *k*d stabla nivoa $2, 3$
- Odabir koordinatne ose na koju je upravna ravan razdvajanja pravi se na osnovu podataka
- I dalje postoji veliki broj mogućnosti za odabir ravni razdvajanja

*k*d stabla – konstrukcija

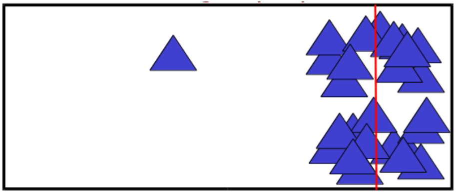
- Postoje različite politike odabira ravni razdvajanja
- Podela po sredini opsega



- Jednaka je verovatnoća da zrak uđe u levi i desni opseg, ali je cena ulaska zraka u desni opseg mnogo veća
- Kako da ovo popravimo?

*k*d stabla – konstrukcija

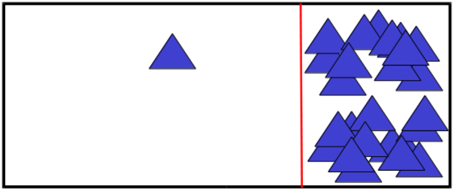
- Podela prema medijani



- Cena ulaska zraka u svaki od opsega je otprilike jednaka, ali je verovatnoća ulaska zraka u levi opseg mnogo veća

*k*d stabla – konstrukcija

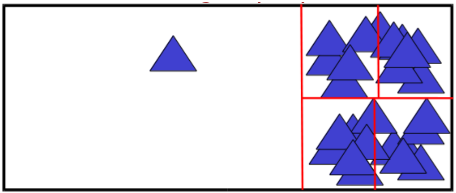
- Podela sa optimalnom cenom



- Pokušavamo da donekle uravnotežimo cenu ulaska u opseg sa verovatnoćom ulaska u taj opseg
- Na ovaj način kreiraju se veliki prazni čvorovi koji se brzo mogu odbaciti tokom obilaska

*k*d stabla – konstrukcija

- Podela sa optimalnom cenom (naredni korak)



- Levi opseg ne zahteva ponovnu podelu, desni delimo oko sredine opsega/prema medijani
- Prednost: izolovali smo jedan geometrijski objekat nakon samo jedne podele

SAH heuristika za izbor ravni razdvajanja

- Pretpostavljamo da su zraci ravnomerno raspodeljeni u prostoru
- Kao moguće ravni razdvajanja razmatramo samo podele duž koordinatnih osa koje sadrže ivice primitiva
- Cenu date ravni razdvajanja računamo kao:

$$c = P_l \cdot c_l + P_d \cdot c_d$$

  gde su P_l i P_d količnici površina (obima) graničnih opsega levog, odnosno desnog deteta i roditelja, a c_l i c_d broj primitiva koje sadrži levi, odnosno desni opseg
- Od svih razmatranih ravni razdvajanja u tekućem čvoru biramo onu koja ima najmanju cenu
- Podelu vršimo samo ako je cena nakon podele manja od cene ulaska u tekući čvor bez dalje podele
- Ovo je heuristika površine površi (eng. surface area heuristic, SAH)

SAH heuristika za izbor ravni razdvajanja

$$c_1 = \frac{18}{36} * 0 + \frac{34}{36} * 3 \approx 2.8$$

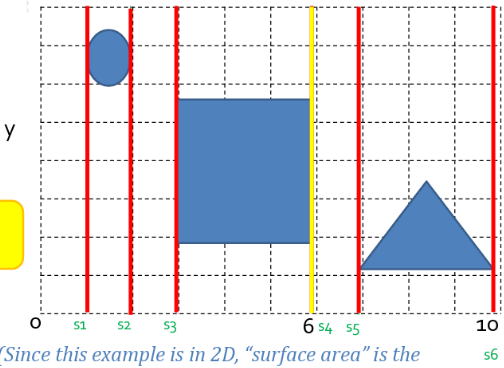
$$c_2 = \frac{20}{36} * 1 + \frac{32}{36} * 2 \approx 2.33$$

$$c_3 = \frac{22}{36} * 1 + \frac{30}{36} * 2 \approx 2.27$$

$$\boxed{c_4 = \frac{28}{36} * 2 + \frac{24}{36} * 1 \approx 2.22}$$

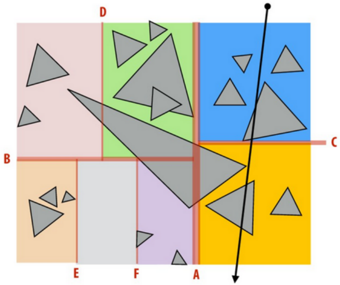
$$c_5 = \frac{30}{36} * 2 + \frac{22}{36} * 1 \approx 2.27$$

$$c_6 = \frac{36}{36} * 3 + \frac{0}{36} * 0 = 3$$



(Since this example is in 2D, "surface area" is the perimeter of each bounding rectangle)

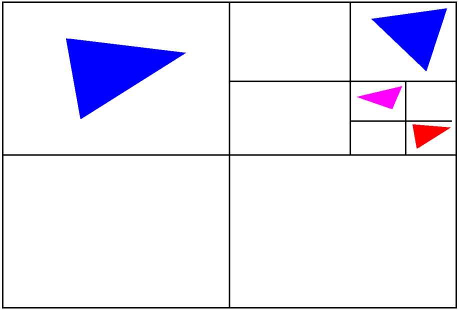
*k*d stablo – upit preseka



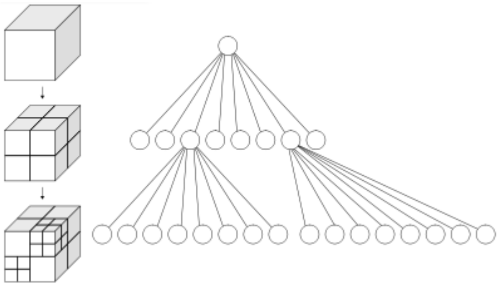
- Čvorove *k*d stabla obilazimo u redosledu kojim ih seče zrak
- Za razliku od hijerarhije graničnih opsega, nakon pronalaska prvog preseka pretraga se može zaustaviti

Oktri i kuodtri

- Možemo izvršiti podelu duž svih koordinatnih osa kroz središte opsega koji zauzimaju primitive u tom čvoru
- Na ovaj način dobija se skup ugnježdenih *k*-kocki
  - u 3D prostoru ovakva struktura podataka naziva se oktri (oct tree)
  - u 2D prostoru ovakva struktura podataka naziva se kuodtri (quad tree)

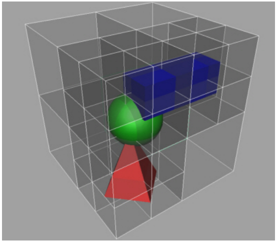


Oktri



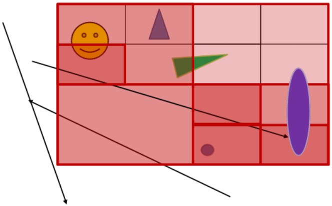
- Najčešće se predstavlja korišćenjem 8 pokazivača ka deci
- Za razliku od uniformne mreže ima veću mogućnost da se prilagode geometriji scene
- Ne moraju sve ćelije biti iste veličine, već se gušći delovi scene sastoje od većeg broja kocki

Oktri (kuodtri) – konstrukcija



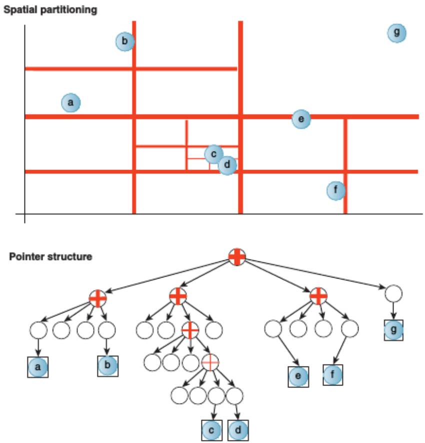
- Konstruiše se odozgo naniže:
    - najpre nalazimo kocku koja ograničava celu scenu – koren stabla
    - u svakoj iteraciji, particionišemo tekući čvor na 8 oktanata i delimo primitive u oktante
    - rekurzivno nastavljamo sve dok oktant ne sadrži dovoljno malo primitiva (ili je dostignut maksimalni dozvoljeni nivo stabla)

Oktri (kuodtri) – upit preseka



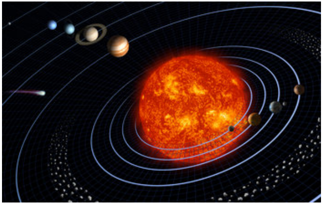
- Ispitivanje preseka počinje u korenu oktrija
- Ako je čvor list, ispitujemo da li zrak seče primitive iz tog čvora
- Ako čvor nije list, iteriramo kroz decu tog čvora u rastućem redosledu rastojanja od početka zraka i proveravamo da li zrak seče ćeliju
  - Ako postoji presek zraka i ćelije, nastavljamo rekurzivno na tom detetu
  - Ako presek ne postoji, prekidamo obradu podstabla u tom čvoru
- Ukoliko postoji presek ranijeg deteta koji je pre preseka sa graničnim opsegom kasnijeg deteta, nema potrebe obilaziti kasnije dete

Kuodtri



Oktri – diskusija

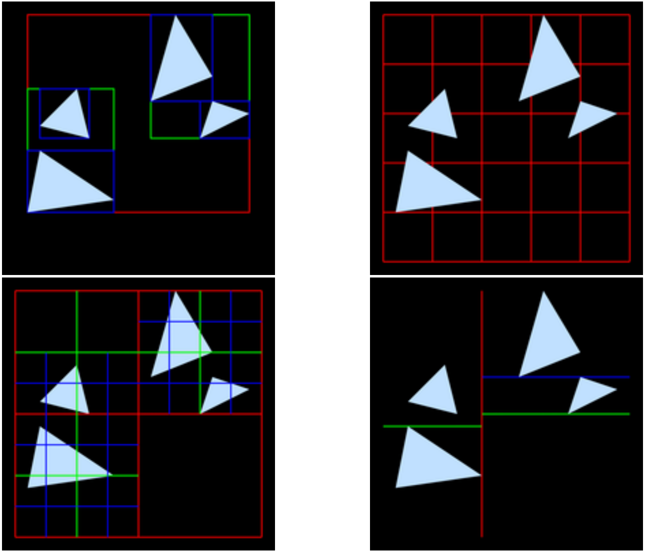
- Očekivano ponašanje oktrija je $O(\log N)$ po zraku
- Međutim, za scene kod kojih su primitive vrlo neravnomerno raspodeljene, oktri će imati jako loše performanse
- Kod ovakvih scena, kd stabla imaju bolje performanse jer se kod njih izoluju delovi scene koji imaju visoku složenost od velikih praznih prostora kroz koje se može brzo proći



Prostorne strukture podataka (rekapitulacija)

- Hijerarhija graničnih opsega
- Mreža
- BSP stabla
- *k*d stabla
- Oktri

Pregled prostornih struktura podataka



Kombinovanje različitih prostornih struktura podataka

- Moguće je iskombinovati različite prostorne strukture podataka
- Na primer, na višem nivou imamo uniformnu mrežu, a u svakoj od ćelija mreže imamo hijerarhiju graničnih opsega