



SVEUČILIŠTE U ZAGREBU



Fakultet
elektrotehnike i
računarstva

Diplomski studij

Računarstvo

Akademска година
2022/2023

Umrežene igre

Arhitekture videoigara i mehanizmi za
skrivanje mrežnog kašnjenja



Prije uvoda

- Prilika za posao u našoj industriji igara
- Overseer Games (bivši LGM) traži testera za igru Aquatico (city builder žanr)
https://www.youtube.com/watch?v=bfmkQ6IJ-w8&t=7s&fbclid=IwAR2Etk3Z7TtNA_PxxMbm8uc2PEGGogOg_ibxdI7FID4i3HApKcD-j6eiU1o&ab_channel=OverseerPublishing
- Dva tjedna intenzivnog testiranja (igranja)
- Posao je u uredu studija u Zagrebu
- Studentski ugovor
- Zainteresirani se mogu javiti na: mario@inter-corona.com
- Dobar izvor informacija o našoj industriji igara je Facebook grupa Game Developers Croatia te se možete učlaniti u nju

Projekt - promjena

- U okviru posljednjeg termina predavanja bit će prezentacije projekata (osvježen je Syllabus)
- Svaki tim treba pripremiti prezentaciju u trajanju od maksimalno 10 minuta
 - Unutar prezentacije treba prezentirati sve što je obećano u planu projekta te povući paralelu s onim što je ostvareno
 - Idealno bi bilo u sklopu prezentacije kratko prikazati igranje igre
 - Igranje bi trebalo biti na stvarnoj mreži (lokalni WiFi StreamsLab – lozinka Rovkp2016!)
 - U krajnjem slučaju može se prikazati video igranja
 - Prezentacija se boduje za projekt!

Sadržaj

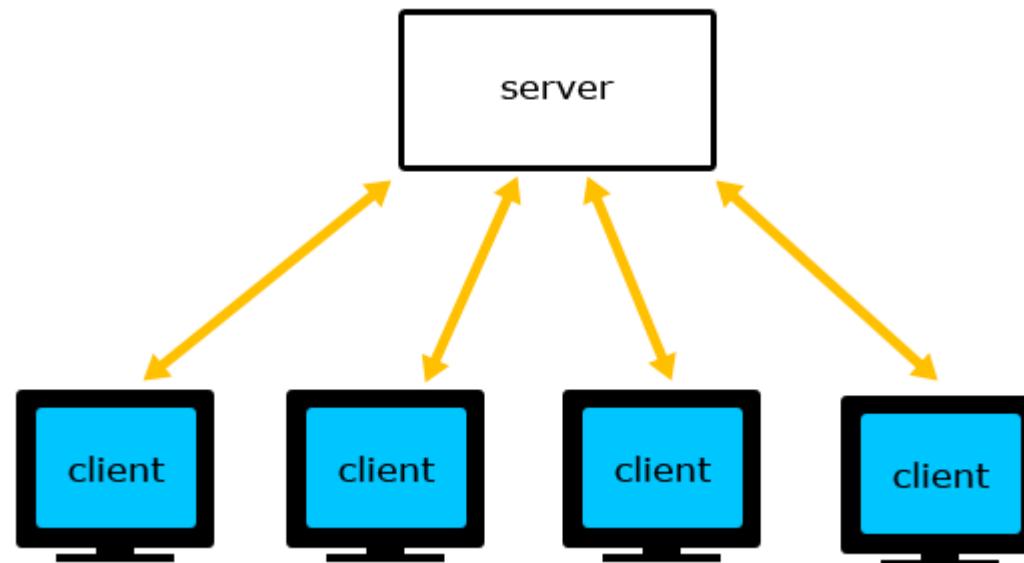
- Arhitekture igara
- Mehanizmi za sakrivanje kašnjenja
 - Mehanizmi na strani poslužitelja
 - Mehanizmi na strani klijenta

Arhitekture videoigara

- Kada govorimo o arhitekturi videoigara možemo temeljiti raspravu na više pogleda
 - Kako su organizirani entiteti distribuiranog sustava koji razmjenjuju informacije?
 - Klijent – poslužitelj
 - Ravnopravni entiteti
 - Tko su vlasnici infrastrukture entiteta koji razmjenjuju informacije?
 - Izdavač/razvijatelj igre
 - Igrači
 - Kakav tip informacija se u sustavu razmjenjuje?
 - Osvježenja i kontrole
 - Video strujanje (i grafičke informacije) i kontrole

Klijent – poslužitelj

- Izvedba usluge u modelu klijent/poslužitelj podijeljena je između programa klijenta i programa poslužitelja, smještenih na krajnjam sustavima spojenima na Internet
 - Koristi se u većini "standardnih" internetskih usluga (web, e-mail, itd.)
 - Jedan poslužitelj poslužuje više klijenata
 - Komunikacija se temelji na nizu zahtjeva i odgovora:
 - Klijent inicira uslugu slanjem zahtjeva poslužitelju
 - Poslužitelj obrađuje zahtjev i odgovara klijentu šaljući rezultat obrade

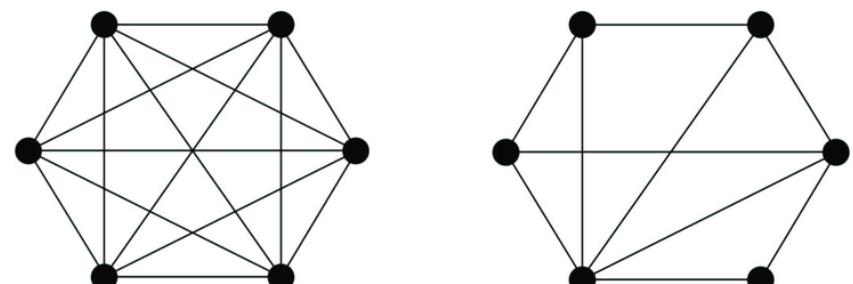
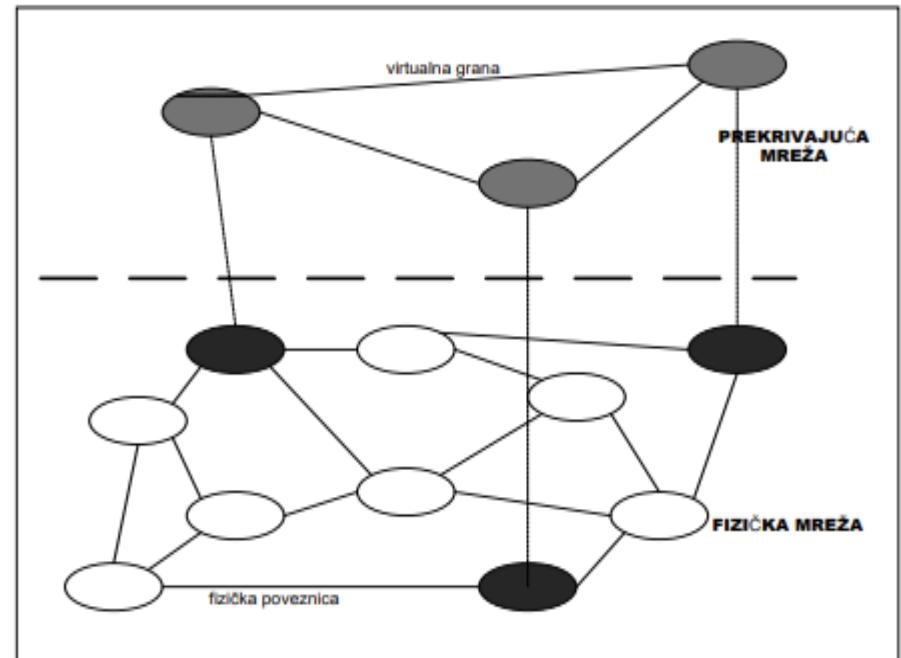


Klijent – poslužitelj

- Prednosti
 - Problemi varanja se lakše riješe– poslužitelj može provjeravati i autorizirati aktivnosti
 - Lakše održavanje konzistentnosti zajedničkog stanja – postoji jedna točka koja rješava i odlučuje o tome koja je „točna“ verzija virtualnog svijeta
 - Uspostava i održavanje mrežne veze – puno lakše za implementirati u sustavu klijent – poslužitelj
 - Sakrivanje distribuirane prirode sustava – lakša implementacija mehanizama za sakrivanje utjecaja mrežnih parametara
 - Napredni poslovni modeli – klijent poslužitelj olakšava implementaciju pretplatničkog ili modela besplatnog igranja s mikro transakcijama
- Mane
 - Poslužitelj može biti usko grlo – preopterećenje poslužitelja uzrokuje degradaciju kvalitete
 - Cijena – održavanje i/ili iznajmljivanje poslužitelja može biti skupo

Ravnopravni entiteti

- U izvedbi ovog modela svi entiteti su ravnopravni te svi implementiraju funkcije i klijenta i poslužitelja, odnosno mogu slati zahtjeve i odgovarati na njih
 - Prekrivajuća mrežna topologija (engl. overlay network) – nastaje kada se entiteti povežu na aplikacijskom protokolu te je postavljena nad stvarnom mrežnom topologijom
 - Prekrivajuća mrežna topologija može biti
 - Potpuno povezana mreža (engl. full mesh) – mreža u kojoj je svaki entitet povezan s drugim
 - Djelomično povezana – mreža u kojoj nije svaki entitet direktno povezan s drugim već poveznica ide preko drugih entiteta
 - Koristi se u mnogim distribuiranim sustavima, a najpoznatiji su bili sustavi za dijeljenje datoteka, a danas se u igrama učestalo koriste za distribuciju osvježavanje verzija igara
 - Rijetko se koristi za distribuciju informacija u samom procesu igranja (primjerice Demigod)

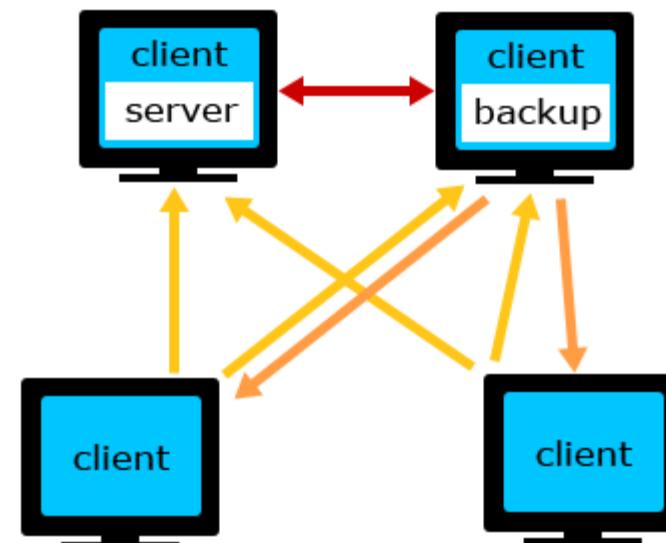
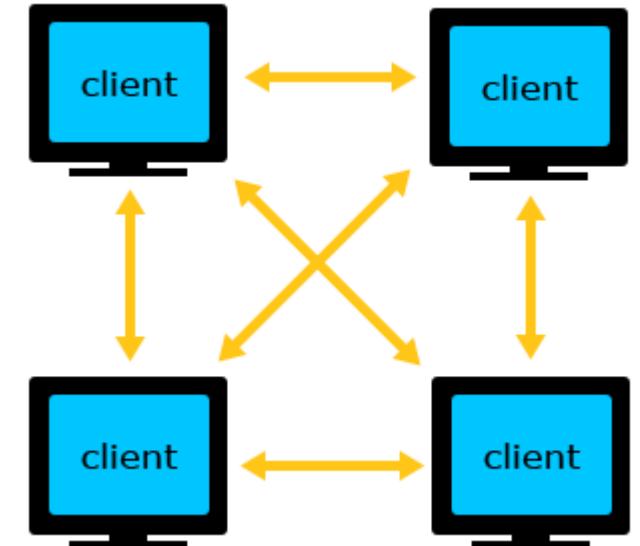


Izvor: Raspodijeljeni sustavi Radna inačica udžbenika v.1.3

https://www.fer.unizg.hr/_download/repository/Rassus-2016_udzbenik_v_1_3.pdf

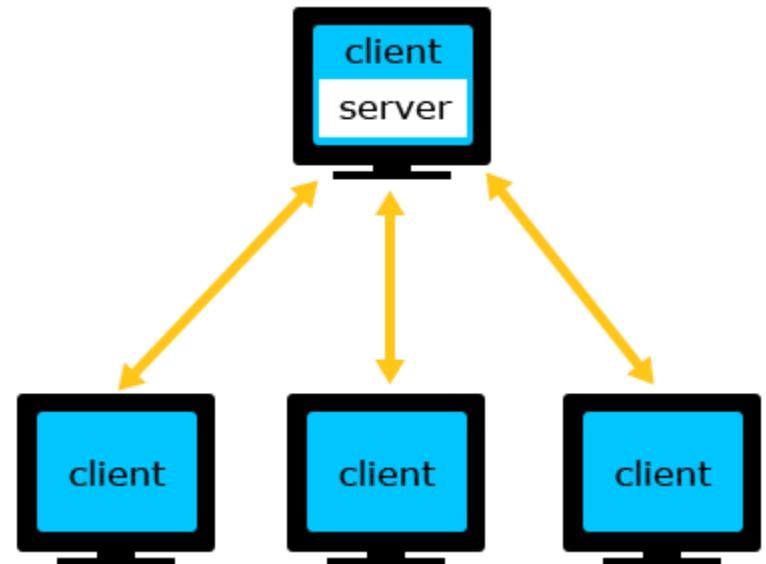
Ravnopravni entiteti

- Prednosti
 - Cijena – ne postoji trošak za izdavača/razvijatelja
 - Skalabilnost – moguće vrlo lako skalirati sustav za veliki broj igrača jer se igra odvija na njihovim računalima
- Mane
 - Varanje – olakšano varanje jer svaki entitet ima pristup svim podacima
 - Nekonzistentnosti – otežano održavanje konzistentnosti, te se ponekad kreiraju entiteti koji imaju veća ovlaštenja od drugih odnosno ovlaštenja slična poslužitelju (engl. super-peer). Primjerice Hydra sustav za razvoj P2P aplikacija koja je ilustrirana na donjoj slici
 - Dinamičnost mrežne topologije (nestabilnost) – pojedini igrači mogu izaći iz sustava pa je potrebno prebacivanje njihovog stanja, poveznica i slično
 - Poslovni modeli – otežane mikrotransakcije (primjerice, ako klijent ima sve informacije o svim mogućim izgledima, može varati i prikazati da ima nekakav izgled koji nije kupio)
 - Kašnjenje – često klijenti nemaju dovoljno veliku brzinu pristupne mreže (pogotovo u uplinku) pa može doći do dodatnog kašnjenja



Tko je vlasnik infrastrukture?

- U arhitekturi ravnopravnih entiteta uvijek su vlasnici infrastrukture igrači
- U klijent – poslužitelj arhitekturi mogu biti oba slučaja
- Poslužitelji na strani izdavača/razvijatelja
 - Vrlo česta opcija u današnjim igrama
 - Primjeri igara:
 - League of Legends
 - World of Warcraft
 - ...
- Poslužitelji na strani klijenta
 - Poslužitelj na klijentskoj strani može biti integriran u klijentsku aplikaciju (primjerice Civilization 6) ili posebna aplikacija koja dolazi s klijentom (primjerice Call of Duty)
 - Često se ova arhitektura naziva i klijent kao poslužitelj, super-peer ili čak ravnopravni entiteti (eng. Peer to peer) iako zapravo to nije P2P arhitektura
 - Ova arhitektura kombinira prednosti arhitekture klijent – poslužitelj i arhitekture ravnopravnih entiteta (troškovi, olakšano održavanje konzistentnosti), ali zadržava i neke mane (poslužitelj je usko grlo pogotovo ako imamo preveliki broj igrača i drugih entiteta u igri, dinamičnost i slično)
 - Vrlo česta opcija u današnjim igrama
 - Primjeri igara
 - Red Dead Redemption 2
 - Grand Theft Auto Online
 - ...
- | Red Dead Redemption 2 i GTA Online koriste Akamai Content Deliver Network (CDN) da bi bolje povezali igrače, odnosno da bi oni imali bolje iskustvo igranja

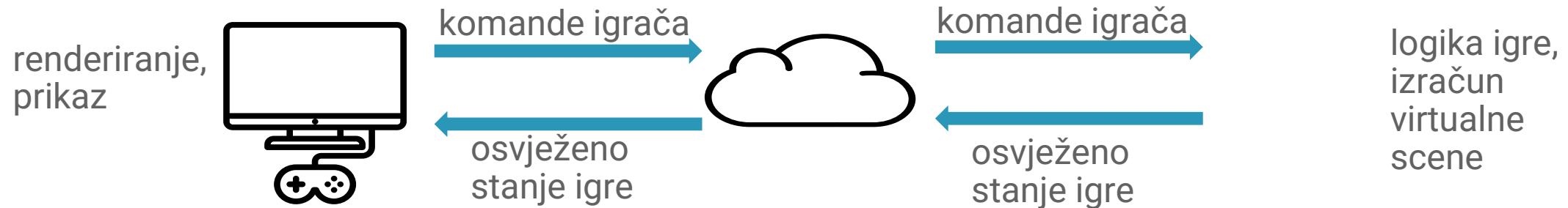


Kakav se tip informacija razmjenjuje

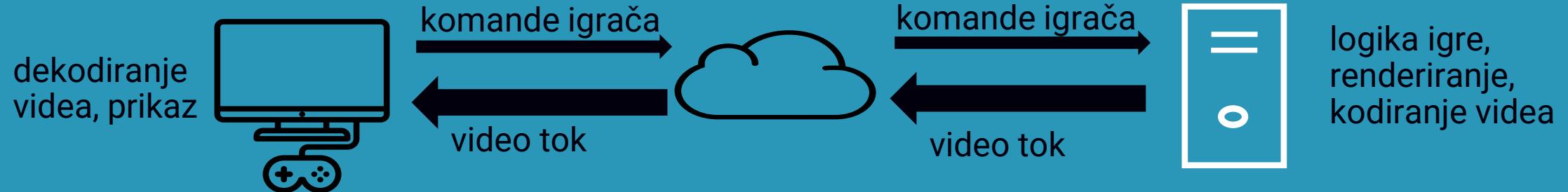
- „Tradicionalni pristup“ – razmjenjuju se komande te osvježenja stanja ili snimci stanja
 - Podrazumijeva posebnu aplikaciju na klijentskoj i poslužiteljskoj strani za svaku igru
 - Mali mrežni zahtjevi
 - Moguće i na arhitekturi ravnopravnih entiteta i na arhitekturi klijent - poslužitelj
- Igre temeljene na računalnom oblaku (engl. cloud gaming)
 - Izračun stanja i renderiranje virtualne scene se događa na poslužitelju
 - Klijentu se šalje strujanje videa visoke razlučivosti
 - Na klijentskoj strani postoji jedna aplikacija kojom se igra više igara
 - Moguće samo u klijent poslužitelj arhitekturi u obje opcije
 - Poslužitelj je na strani pružatelja usluge – primjerice GeForce Now
 - Poslužitelj je na strani klijenta – primjerice Steam Remote PLay

Kakav se tip informacija razmjenjuje

Platforma: tradicionalna mrežna igra



Igre zasnovane na računalnom oblaku

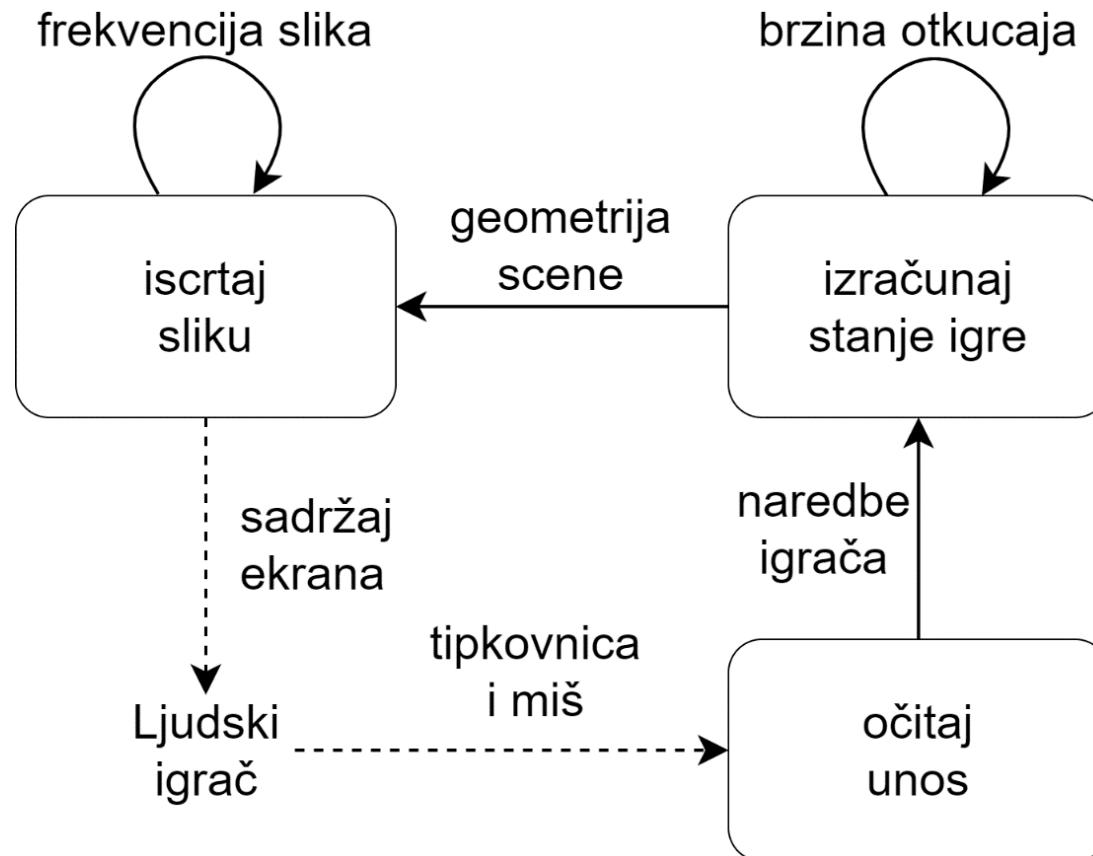


Kakav se tip informacija razmjenjuje?

- Prednosti tradicionalnog pristupa
 - Niz mogućih optimizacija na strani klijenta i poslužitelja, primarno za sakrivanje kašnjenja
 - Malo korištenje mrežnog prometa
- Mane tradicionalnog pristupa
 - Svaka igra ima svog klijenta te zauzima prostor na disku
 - Potrebno jako računalo za složenije igre
 - Potrebna aplikacija posebna za svaku platformu – jako velik dodatni trošak razvoja
- Prednosti igara temeljenih na računalnom oblaku
 - Jedan klijent za sve igre – malo zauzeće memorije
 - Nije potrebno snažno računalo za igranje
 - Nije potrebno osvježavanje klijenata igre
 - Lakše sa razvoj – jedna aplikacija za sve platforme
 - Pojačana sigurnost – gotovo nemoguće varanje
- Mane igara temeljenih na računalnom oblaku
 - Potrebna vrlo dobra mrežna konekcija i velika potrošnja prometa – problematično za mobilne korisnike pa čak i u Južnoj Koreji gdje su telekomi pokušali potaknuti korisnike da uzimaju uslugu igara temeljenih na računalnom oblaku, ali bez većeg tržišnog uspjeha
 - Veća osjetljivost na kašnjenje
 - Osjetna degradacija kvalitete videa kod degradacije mrežne usluge
 - Poslovni modeli distribucije igara – ako imate igru na Steamu, zašto bi je opet kupili na GeForce Now platformi?
 - Još uvijek nisu široko prihvaćeni na tržištu

Osnovna petlja funkcioniranja igre

- Prikazivanje slike korisniku se radi određeni broj puta u sekundi.
- Standard visoke kvalitete korisničkog iskustva zahtjeva 60 sličica u sekundi (engl. Frame Per Second skr. FPS).
- Korisnička istraživanja su pokazala da igrači moguigrati s visokom kvalitetom i s brzinama između 30 i 60 sličica u sekundi.
- Sveukupno percipirano korisničko kašnjenje od točke pritiska gumba do točke u kojoj se vidi rezultat te na njihovim ekranima (uz pretpostavku da nemamo predikciju na strani klijenta) ovisi o brzini osvježavanja sličica na prikazu korisnika, mrežnom kašnjenju i broju otkucaja koji postoji na poslužitelju.
- Osnovna petlja izvršavanja funkcija unutar videoigre može izgledati kako je prikazano na slici
- Isti koncept se lako prenosi na umreženu igru u kojoj je izračun stanja igre samo na udaljenom poslužitelju



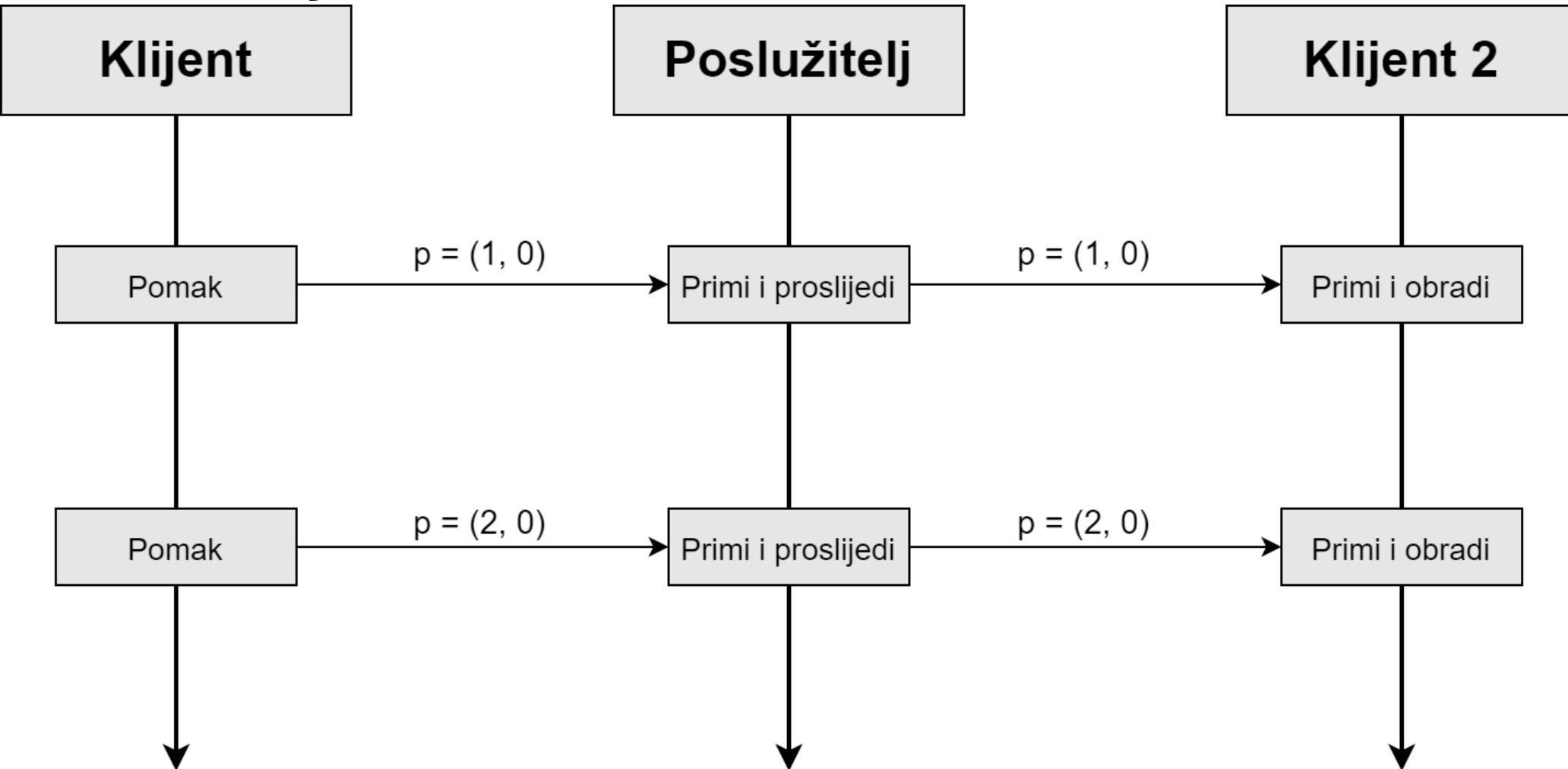
Poslužitelj u klijent – poslužitelj arhitekturi

- Poslužitelj u klijent-poslužitelj sustavu gotovo uvijek ima kompletno zajedničko stanje
- Zajedničko stanje se na poslužitelju obnavlja u diskretnim vremenskim trenutcima unutar simulacije pod nazivom otkucaji (engl. tick)
- Ovo znači da virtualni svijet ne postoji kontinuirano već u diskretnim vremenskim trenutcima, a na klijentu se stanje između tih trenutaka interpolira
- Broj otkucaja u sekundi naziva brzina otkucaja (engl. tickrate).
- Brzine otkucaja variraju o pojedinim tipovima igre kao i o tehničkoj izvedbi igre te one mogu varirati od 20 do čak 128 otkucaja u sekundi
 - Primjerice, Source pokretač igara od razvojnog studija Valve pokreće više igara
 - Counter Strike Source ima brzinu otkucaja od 66, kao i u Team Fortress 2,
 - Brzina otkucaja je 30 za Left 4 Dead i Left 4 Dead 2.
- Izračun pojedinačnog otkucaja se temelji na unosima svih klijenata koji su do tada došli do poslužitelja
- Rezultati svakog otkucaja mogu se, a ne moraju poslati istom brzinom putem mreže
- U pojedinim pokretačima igara ili bibliotekama za umrežavanje igara mogu se postaviti različiti vrijednosti.

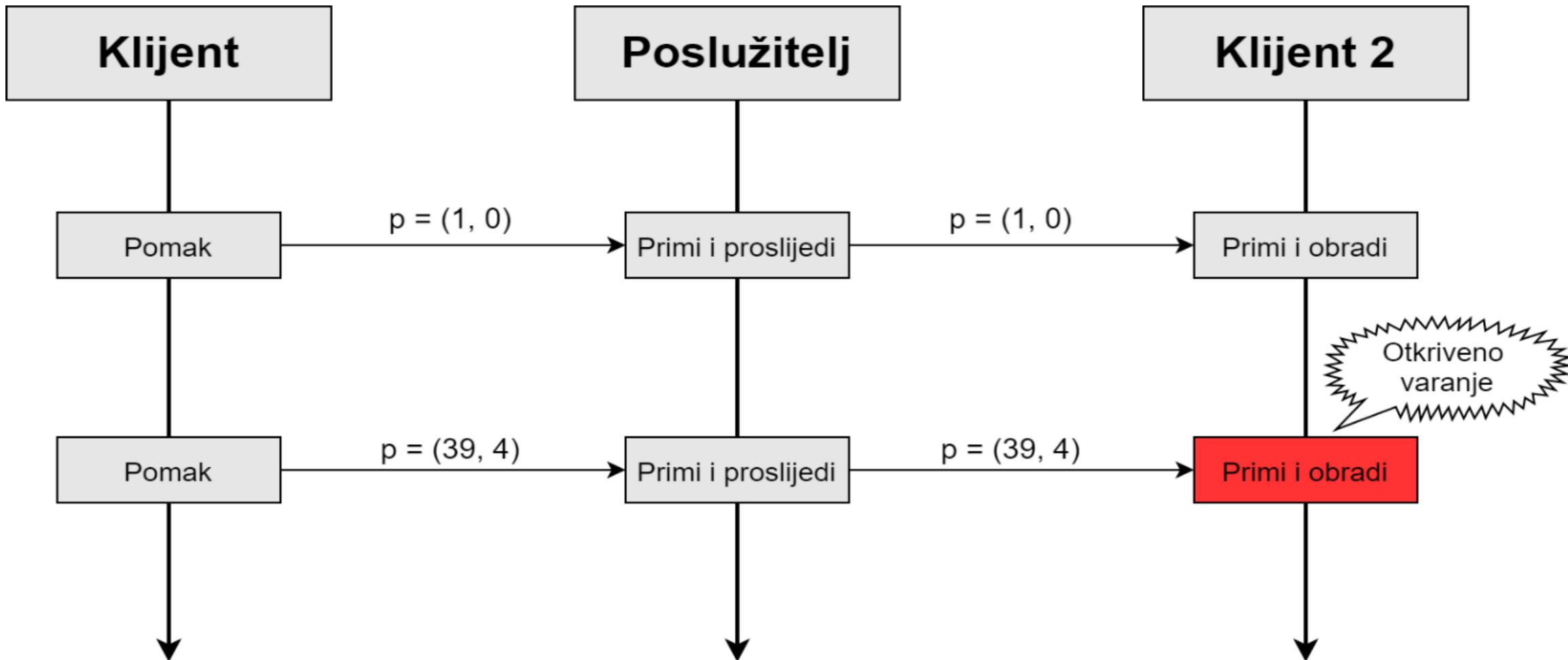
Autoritet i vlasništvo

- Kada nema potpune konzistentnosti dolazi do problema vlasništva nad objektima
- **Vlasništvo** nad objektom u videoigri označava mogućnost mijenjanja stanja tog objekta
- Treba spriječiti da više igrača istovremeno mijenja stanje entiteta Na primjer, promjena položaja je pisanje variable stanja položaja entiteta (x, y, z)
- Uvodi se eksplisitno vlasništvo nad entitetom
 - Tipičan primjer je korisnikov *avatar*, čiji je vlasnik korisnik koji njime upravlja
 - Drugim objektima upravlja poslužitelj *autoritet*
- **Autoritet** označava pravo na donošenje konačne odluku oko promjene stanja nekog objekta u videoigri
 - Osigurava da svaki entitet u zadanim trenutku ima samo jednog vlasnika
 - Osigurava da se klijenti ponašaju po zadanim pravilima, odnosno smanjuje varanje

Poslužitelj bez autoriteta

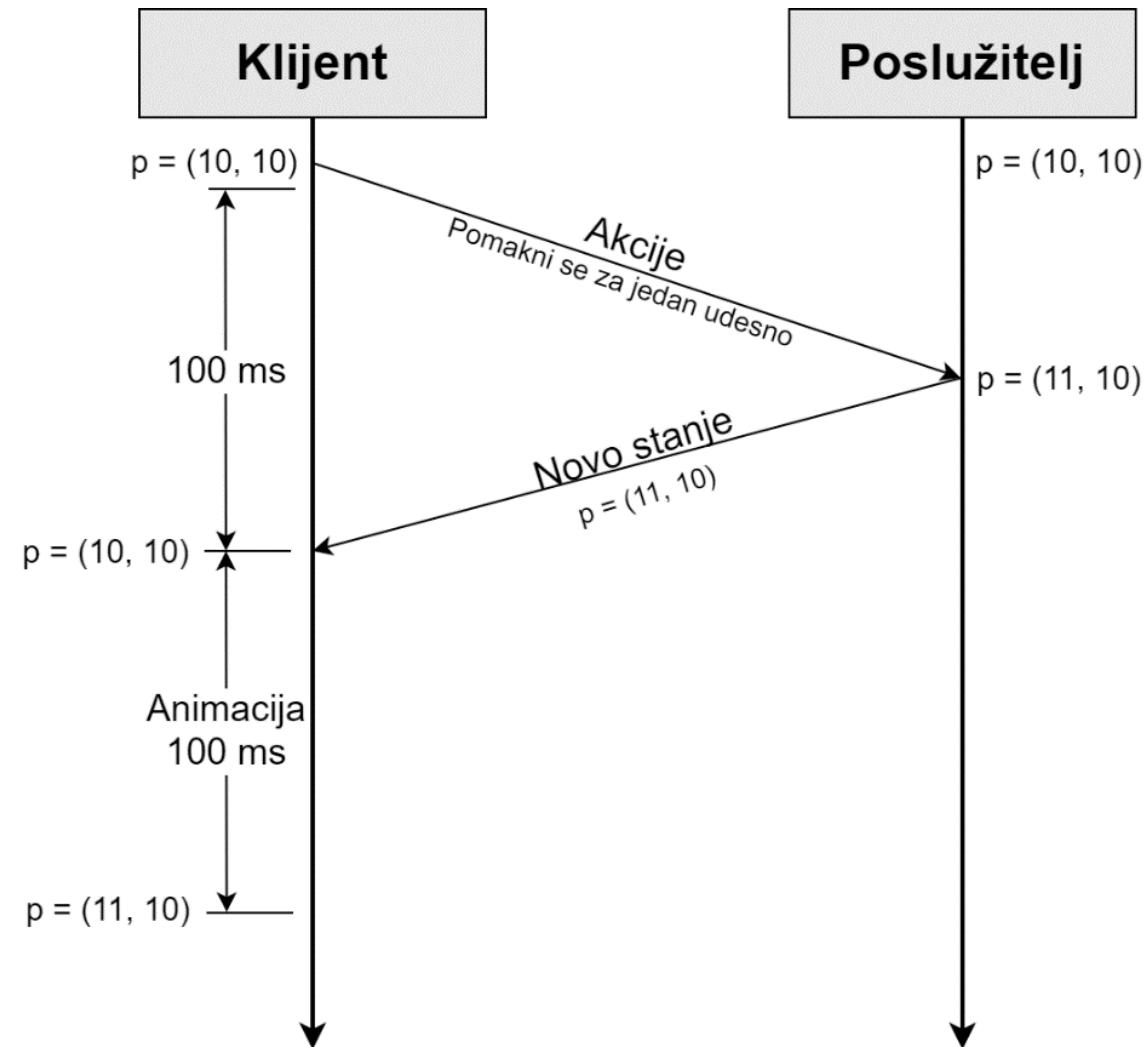


Poslužitelj bez autoriteta - varanje



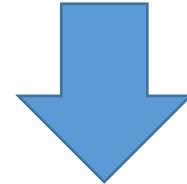
Poslužitelj s autoritetom

- Autorizira svaku komandu klijenta kroz potvrdu u svojoj simulaciji
- U današnjim igrama češće primijenjen model jer je varanje ozbiljan problem
- Ovakav model dodaje kašnjenje
- Protiv kašnjenja se može boriti s tehnikama na strani klijenta te tehnikama na strani poslužitelja
- Ovo specifično kašnjenje se adresira na razini klijenta

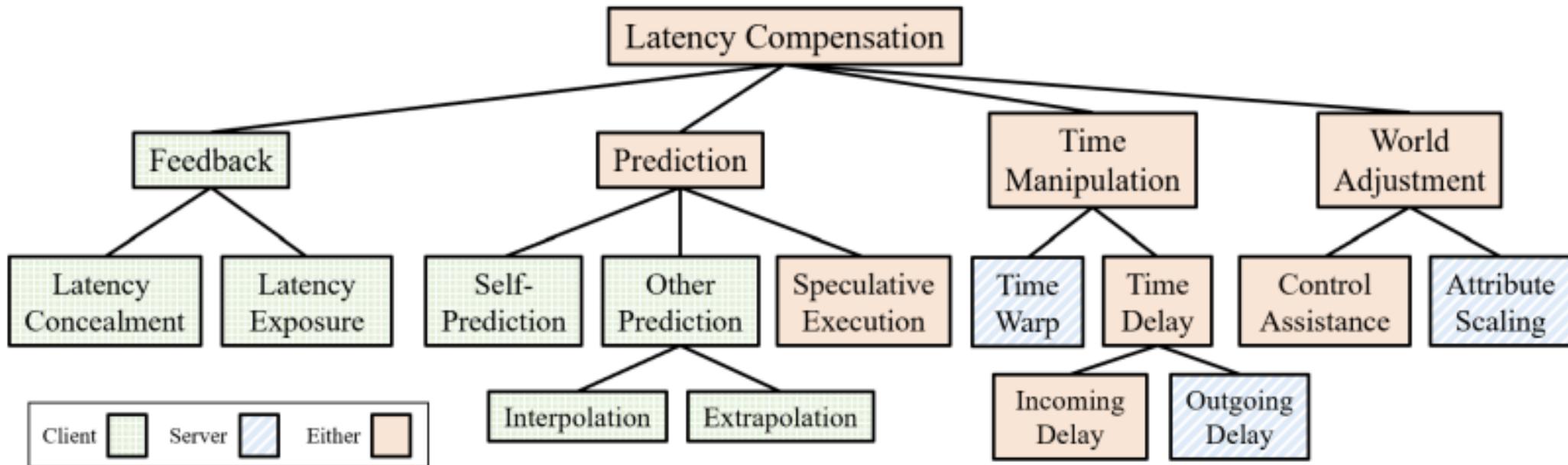


Metode za kompenzaciju kašnjenja

- Kašnjenje može biti vrlo frustrirajuće za igrače umreženih videoigara kao što je pokazani na prethodnom predavanju
- Protiv kašnjenja se na bore **metode za kompenzaciju kašnjenja**
- Na strani poslužitelja
 - Vremenske prilagodbe,
 - Premotavanje vremena,
 - Dodavanje kašnjenja,
 - Prilagodbe virtualnog svijeta,
 - Pomoć s kontrolama,
 - Mijenjanje karakteristika virtualnog svijeta i
 - Spekulativno izvođenje
- Na strani klijenta
 - Sakrivanje kašnjenja
 - Prikaz kašnjenja
 - Predikcija ponašanja igrača
 - Ekstrapolacija ponašanja entiteta
 - Interpolacija ponašanja entiteta



Tehnike za kompenzaciju kašnjenja



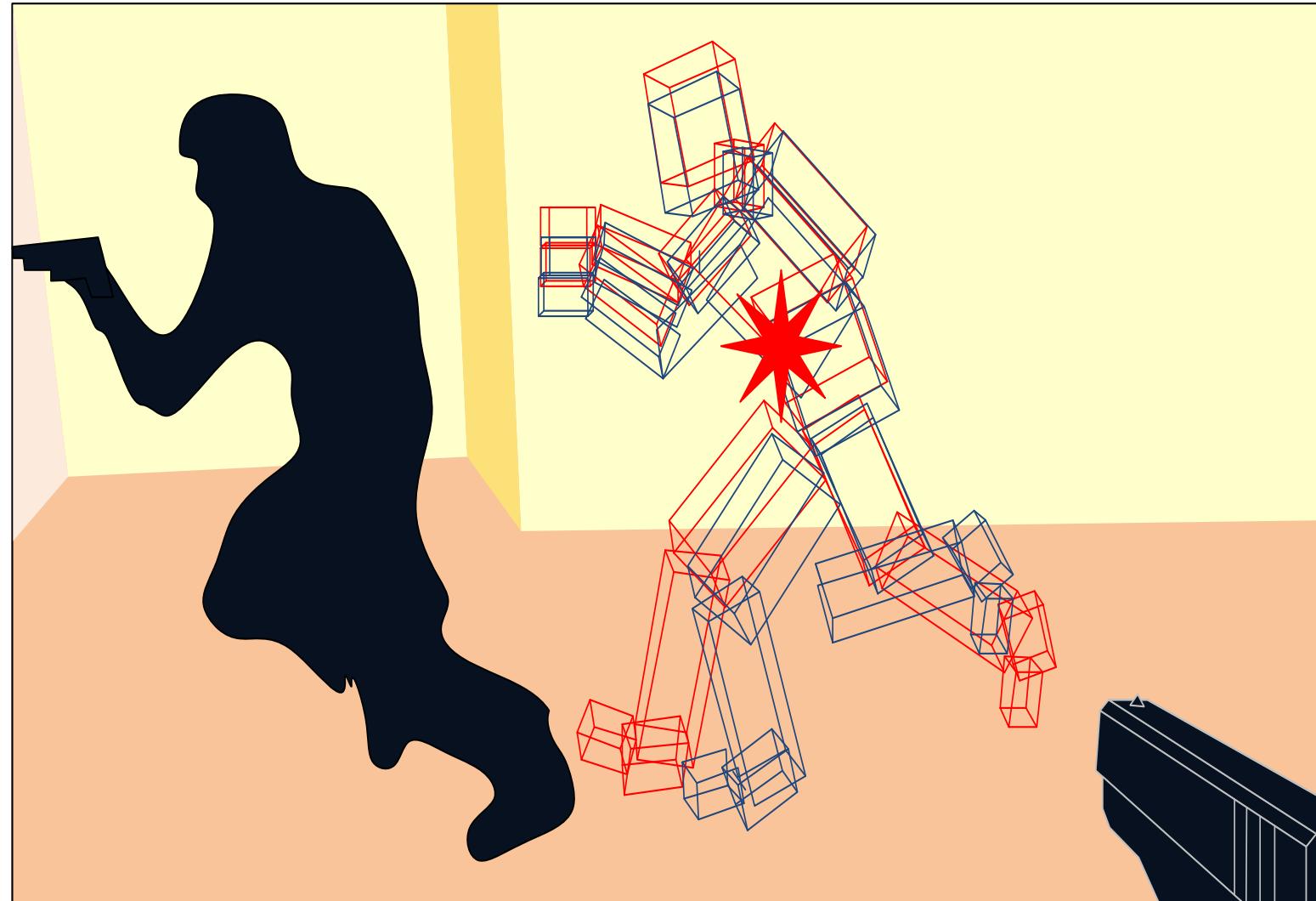
Izvor: Liu, Shengmei, Xiaokun Xu, and Mark Claypool. "A Survey and Taxonomy of Latency Compensation Techniques for Network Computer Games." *ACM Computing Surveys (CSUR)* (2022). <https://ftp.cs.wpi.edu/pub/techreports/pdf/21-06.pdf>

Premotavanje vremena

- **Premotavanje vremena** funkcioniра тако да послуžitelj pohranjuje стање виртуелног свјета неко vrijeme, а потом ауторизира акције клијента на „верзији“ виртуелног свјета коју клијент види, односно која је помакнута у прошlost за мрежно каšnjenje клијента
 - Најчешће се не погодује више од једне секунде
 - 250 ms каšnjenja у случају Overwatcha
 - 500 ms у случају Call of Duty: Infinite Warfare
- Примјер:
 - Ана има каšnjenje од 50 ms, а Божан од 300 ms. Рекомо да Ана претрчава преко празног простора, 53 crnom sjenom је приказано где је Ана заправо на послуžitelju, а crvenim је označено где је kolizijski okvir (engl. hitbox) reprezentације Aninog avatara на Bojanovom računaљу. Bojan naravno гада Aninog avatara тамо где га види. Kad bi послуžitelj uspoređivao poziciju где је Bojan гађао са стварном pozicijom Aninog avatara коју он види Bojan bi promašio! Kada Bojan puca, послуžitelj на темељу njegovog каšnjenja може izračunati približno што Bojan заправо види те је приказано plavim kolizijskim okvirom

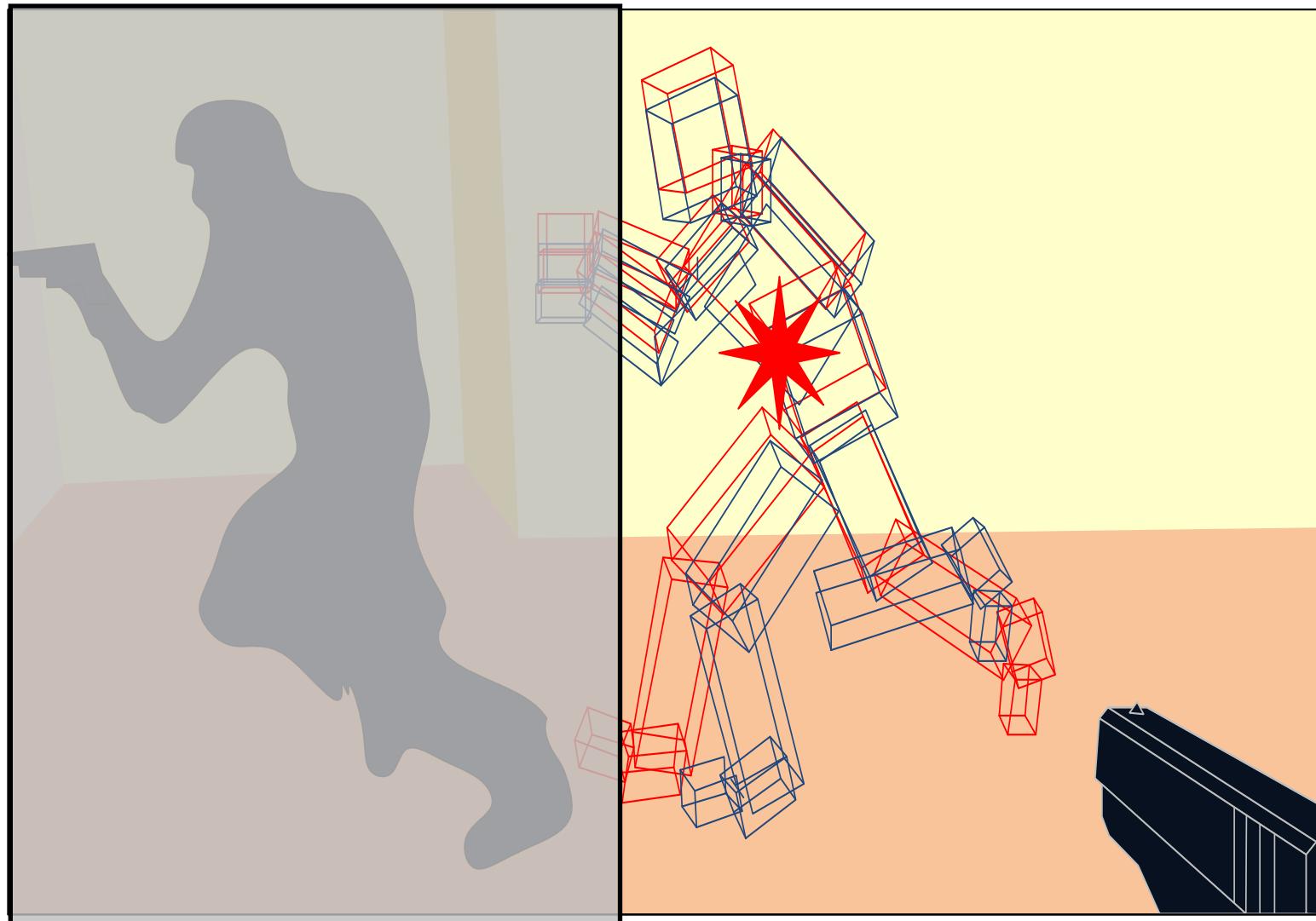
Premotavanje vremena – primjer

- Crna silueta – pozicija Ane na poslužitelju
- Plava silueta – pozicija kolizijskog okvira koju vidi klijent (Bojan)
- Crvena silueta – pozicija kolizijskog okvira kojeg vidi poslužitelj kada premota vrijeme na Bojanov pogled
- Puno veće preklapanje crvenog i plavog kolizijskog okvira nego plavog i crne siluete



Premotavanje vremena – problem obrnute smrti iza zida

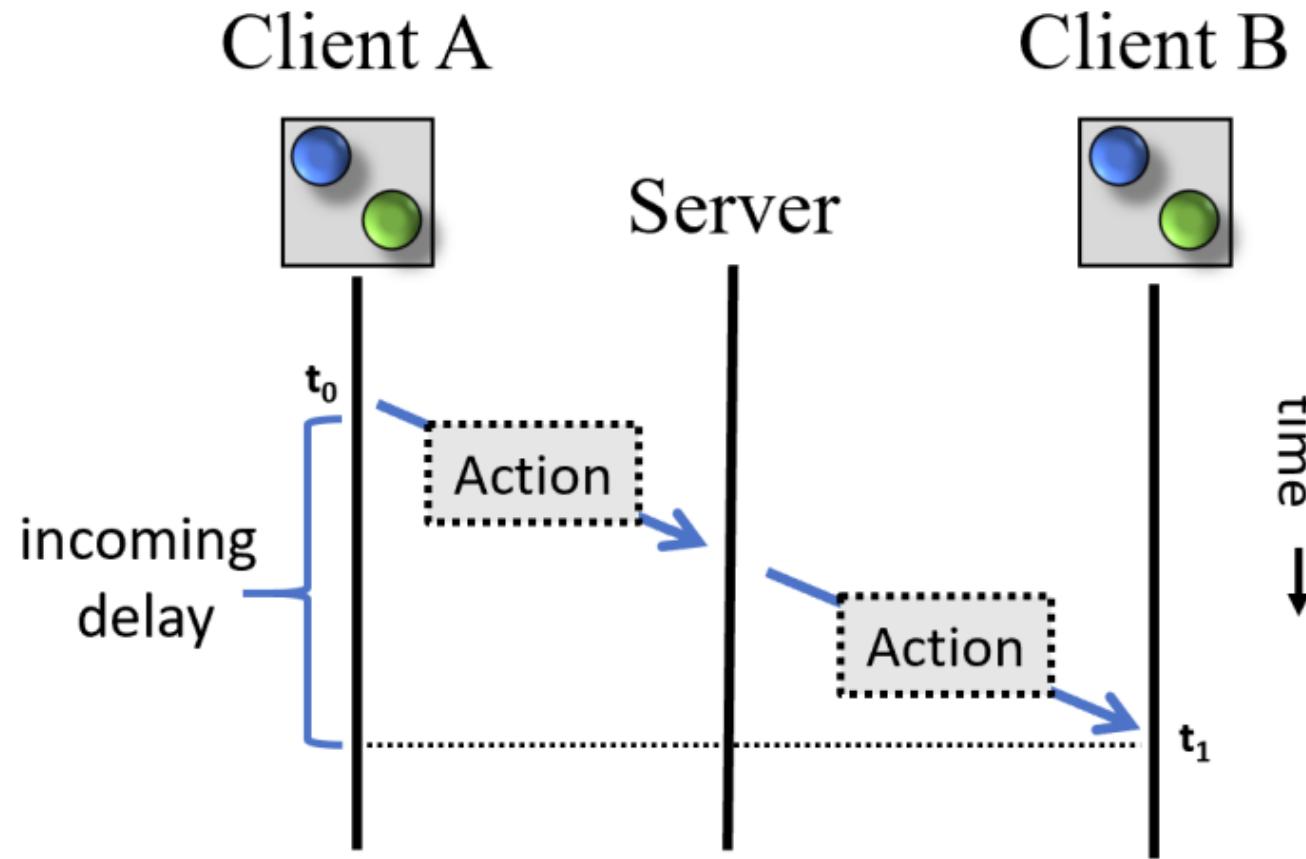
- Sve tehnike kompenzacije kašnjenja imaju i neki kompromis
- Obrnuti problem smrti iza zida
 - Ana se u svojoj simulaciji stigla sakriti iza zida
 - Bojan u svojoj simulaciji još vidi Anu koja je na otvorenom
 - Poslužitelj zna što Bojan vidi te na temelju toga procjenjuje je li Bojan pogodio ili nije
 - Ana pogiba protivno zakonima fizike koje ona vidi u svojoj instanci virtualnog svijeta
 - Sada igrač koji ima manje kašnjenje može biti pogoden iza zida od strane igrača s većim kašnjenjem jer poslužitelj vraća vrijeme u perspektivu igrača s većim kašnjenjem



Dodavanje kašnjenja

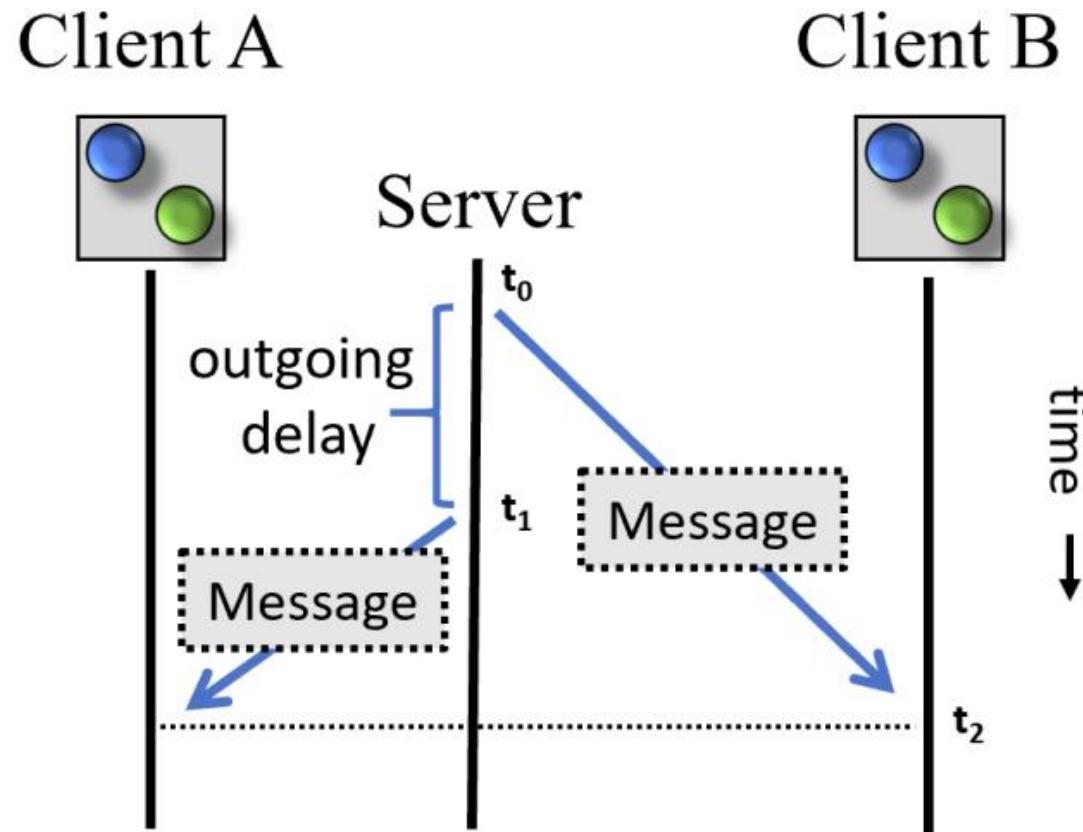
- **Dodavanje kašnjenja** je vrlo jednostavna i služi za ujednačavanje kašnjenja između klijenata kako ne bi bilo razlika koje su zapravo najveći uzročnik različitih nekonzistentnosti
- **Kada svi igrači imaju isto kašnjenje konzistentnost je uvelike povećana**
- Kašnjenje se može dodati na unos od strane klijenata ili osvježenja koje šalje poslužitelj
- Dodavanje kašnjenja može biti na ulazne informacije ili na izlazne informacije
- Prednosti
 - Jednostavan pristup
 - Rješava problem nekonzistentnosti
- Mane
 - Dodaje dodatno kašnjenje
 - Značajno degradira kvalitetu za igrače s brzim vezama, ako je u igri igrač koji ima jako lošu vezu

Kašnjenje u unosu (engl. incoming delay)



Izvor: Liu, Shengmei, Xiaokun Xu, and Mark Claypool. "A Survey and Taxonomy of Latency Compensation Techniques for Network Computer Games." *ACM Computing Surveys (CSUR)* (2022). <https://ftp.cs.wpi.edu/pub/techreports/pdf/21-06.pdf>

Odlazno kašnjenje (engl. outgoing delay)



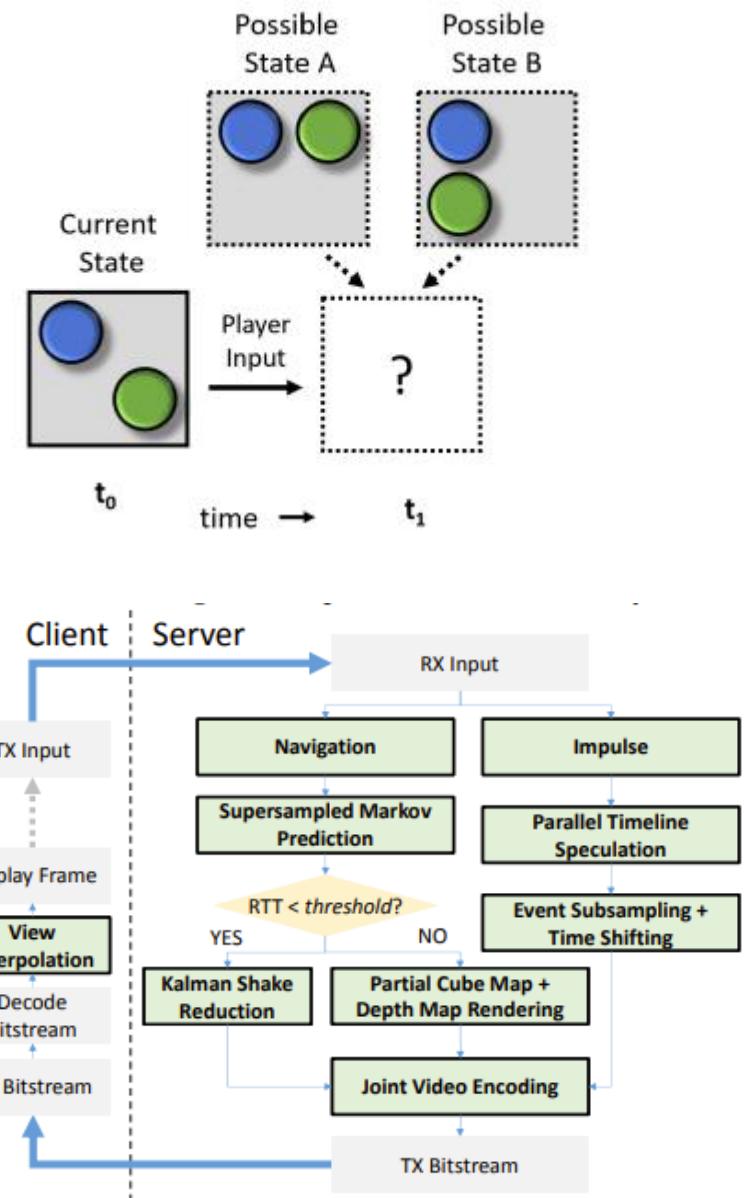
Metode mijenjanja virtualnog svijeta

- **Pomoć s kontrolama je** metoda u kojoj se klijentima koji imaju veće kašnjenje može se dodati da iako možda cilja van mete njegov pucanj se registrira kao pogodak u metu
 - Udaljenost od mete na kojoj igrač dobiva pomoć može biti proporcionalna iznosu kašnjenja.
 - Pomoć s kontrolama se često koristi kod mobilnih verzija igara gađanja kao što je primjerice Fortnite.
- **Mijenjanje karakteristika virtualnog svijeta je** metoda u kojoj se mijenjaju karakteristike virtualnog svijeta, odnosno olakšavaju ograničenja u slučaju da igrači imaju veće kašnjenje
 - Primjerice u igri u kojoj igrač mora proći kroz labirint bez dodirivanja zidova, za igrače koji imaju veće kašnjenje mogu se povećati širine prolaza kako bi se igraču pomoglo da prijeđe igru.



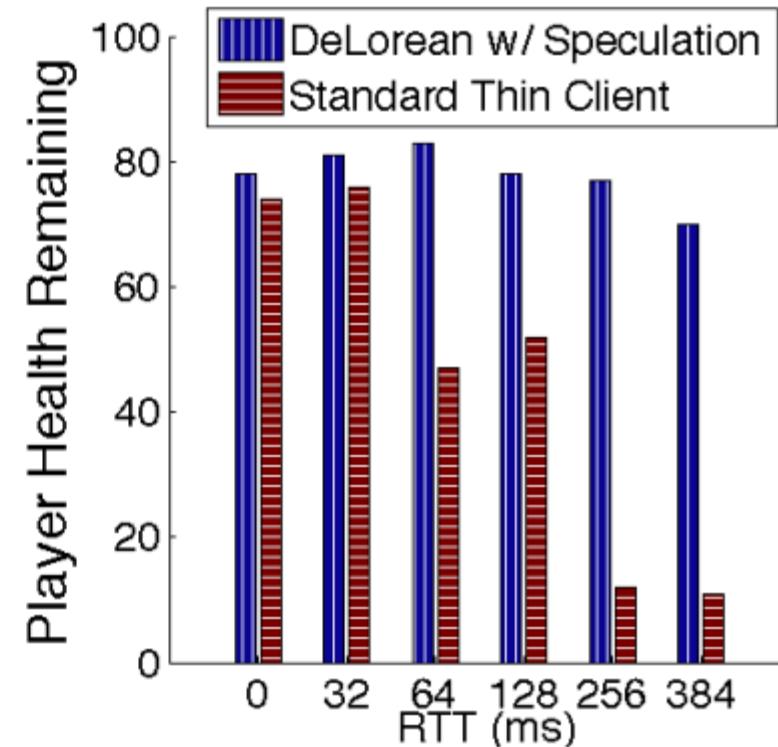
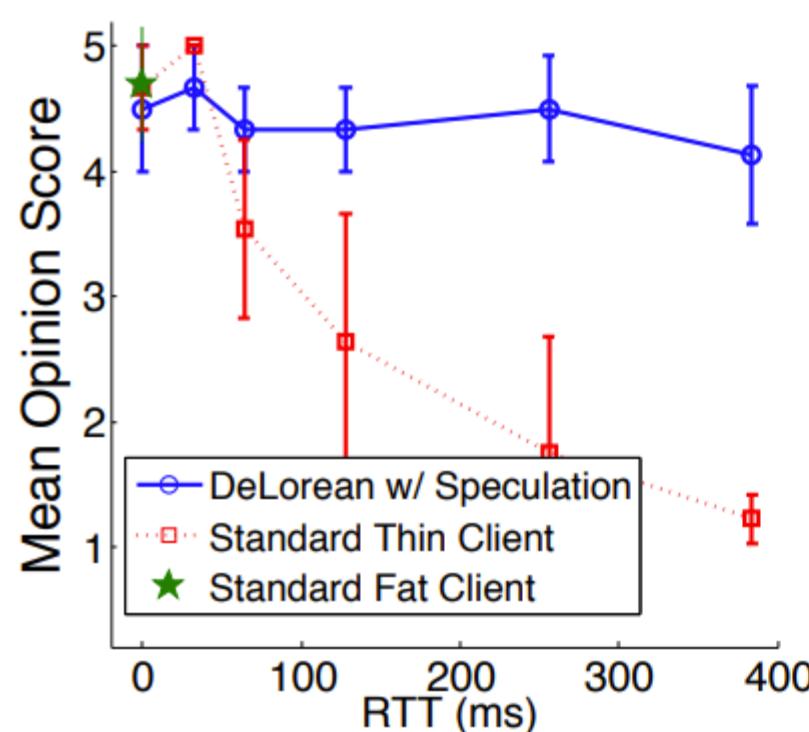
Špekulativno izvođenje

- Pokušava pogoditi kako će svijet izgledati nakon korisničkog unosa i unaprijed šalje različite verzije rezultata
- Ovime se nakon korisničkog unosa može trenutno prikazati rezultat korisničkog unosa bez mrežnog kašnjenja
- Vrlo složen sustav
- Jedna od metoda koja se može primjenjivati i kod igara temeljenih na računalnom oblaku, šalje se jedan ili više video okvira temeljem predviđanja unosa korisnika
 - Primjer Outatime sustav: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/outatime_techreport2014.pdf
- Mreža se dodatno opterećuje jer se šalju dodatna stanja odnosno video okviri



Spekulativno izvođenje

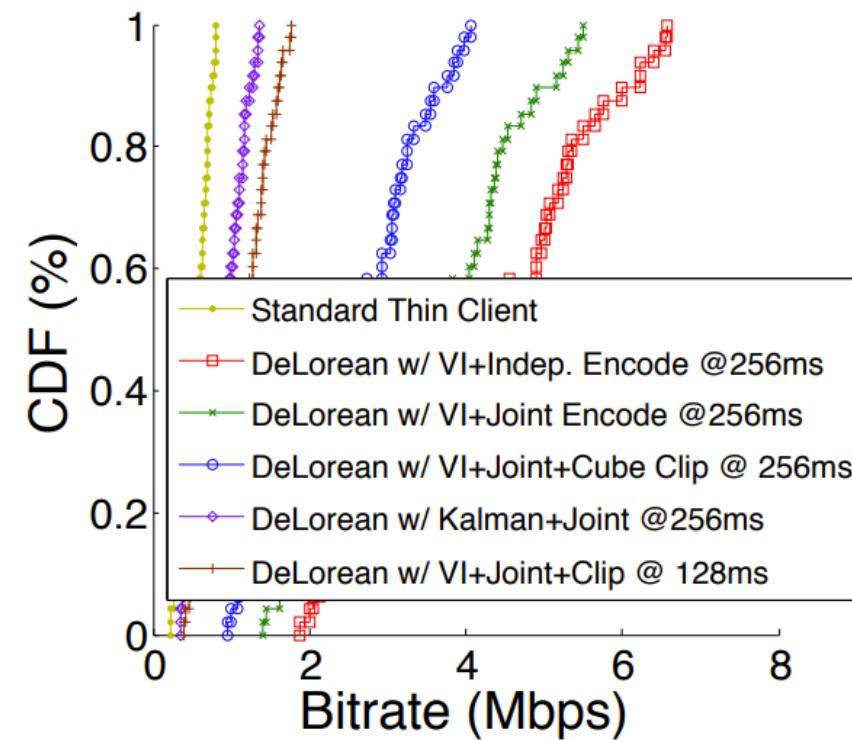
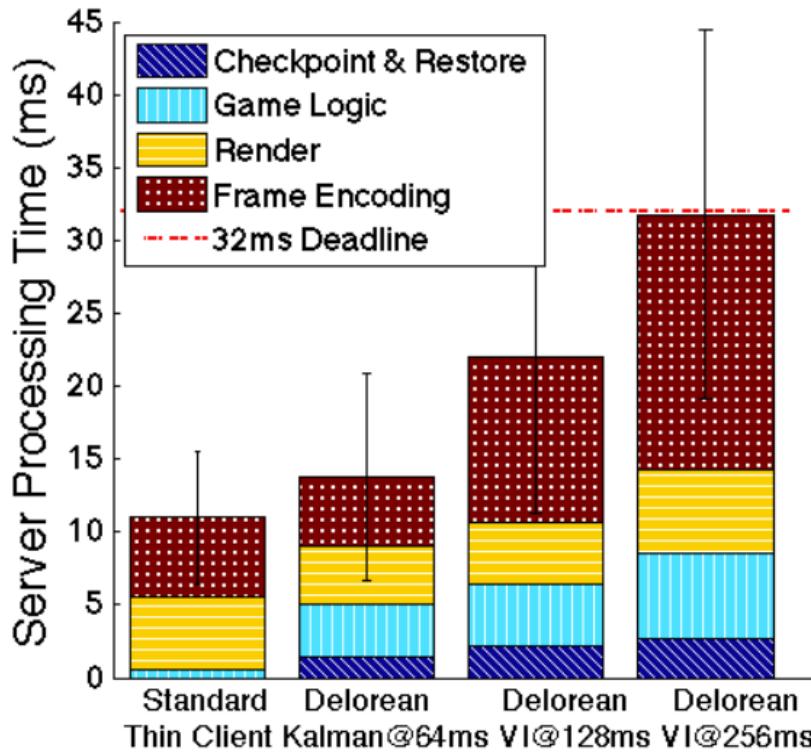
- Značajno bolje objektivne i subjektivne performanse prilikom većeg mrežnog kašnjenja!!!



Izvor: Kyungmin Lee, David Chu, Eduardo Cuervo, Johannes Kopf, Yury Degtyarev, Sergey Grizan, Alec Wolman, and Jason Flinn. 2015. Outatime: Using Speculation to Enable Low-Latency Continuous Interaction for Mobile Cloud Gaming. In Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys). Florence, Italy, 151–165e
<https://ftp.cs.wpi.edu/pub/techreports/pdf/21-06.pdf#page=32&zoom=100,102,169>

Spekulativno izvođenje

- Svaka tehnika ima i negativnu stranu
- Duplo veće kašnjenje procesiranja, te nekoliko puta veće opterećenje mreže u smislu propusnosti



Izvor: Kyungmin Lee, David Chu, Eduardo Cuervo, Johannes Kopf, Yury Degtyarev, Sergey Grizan, Alec Wolman, and Jason Flinn. 2015. Outatime: Using Speculation to Enable Low-Latency Continuous Interaction for Mobile Cloud Gaming. In Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys). Florence, Italy, 151–165e
<https://ftp.cs.wpi.edu/pub/techreports/pdf/21-06.pdf#page=32&zoom=100,102,169>

Klijent u arhitekturi klijent – poslužitelj

- **Osnovne funkcije** na razini klijenta su:
 - Unos komandi i njihovo slanje poslužitelju (isto kao na običnim videoigramama)
 - Izvršavanje klijentske logike
 - Izvođenje simulacije
 - Interpolacija snimaka
 - IsCRTavanje virtualne scene (područje računalne grafike)
 - Prikaz virtualne scene (isto kao na običnim videoigramama)
- Ovisno o odabranom modelu sinkronizacije distribuiranog zajedničkog stanja izvođenje logike videoigre može biti temeljeno na izvođenju simulacije ili interpolaciji snimaka
- Izvođenje simulacije znači da klijent ima mogućnost izračunavati pojedino stanje (njegovog područja interesa) virtualnog svijeta
- Izvođenje simulacije virtualnog svijeta omogućuje implementaciju različitih tehnika za kompenzaciju kašnjenja

Sakrivanje kašnjenja

- Sakrivanje kašnjenja je jednostavna tehnika kojom se **određenim vizualnim efektima sakriva vrijeme koje je potrebno da poslužitelj autorizira nekakvu aktivnost na klijentu**
- Primjerice kada klijent povuče okidač pištolja može se automatski prikazati animacija repetiranja pištolja i bljesak izlaska projektila iz njega što klijentu daje dojam da je odgovor videoigre na komandu trenutan iako poslužitelj još nije potvrdio rezultat te akcije u smislu pogotka određene mete



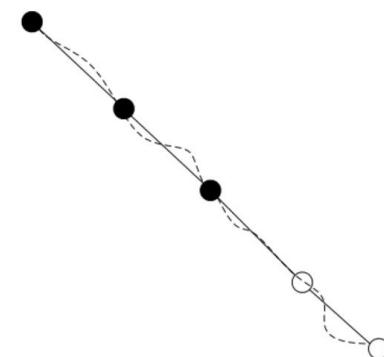
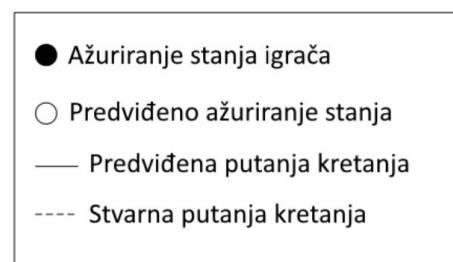
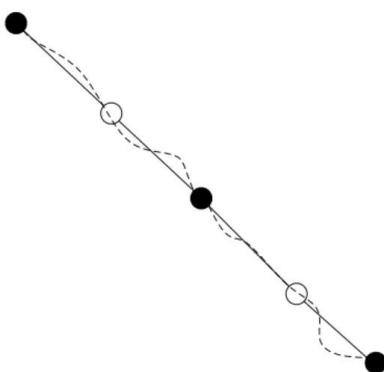
Prikazivanje vrijednosti kašnjenja

- Prikazivanje vrijednosti kašnjenja u obliku broja ili vizualnog indikatora je jednostavna metoda borbe protiv mrežnog kašnjenja koja informiranjem korisnika postiže prilagodbu ponašanja korisnika.
- Nekoliko studija je pokazalo da se performanse igrača mogu poboljšati kada igrači znaju koliko je mrežno kašnjenje pa prema tome prilagode svoju igru
- Studije su pokazale da prilikom pridruživanja poslužiteljima igara gađanja kašnjenje prema poslužitelju je glavni faktor pri odabiru poslužitelja
- Možete li se sjetiti igre gdje je kašnjenje stalno pokazano nekakvim indikatorom?



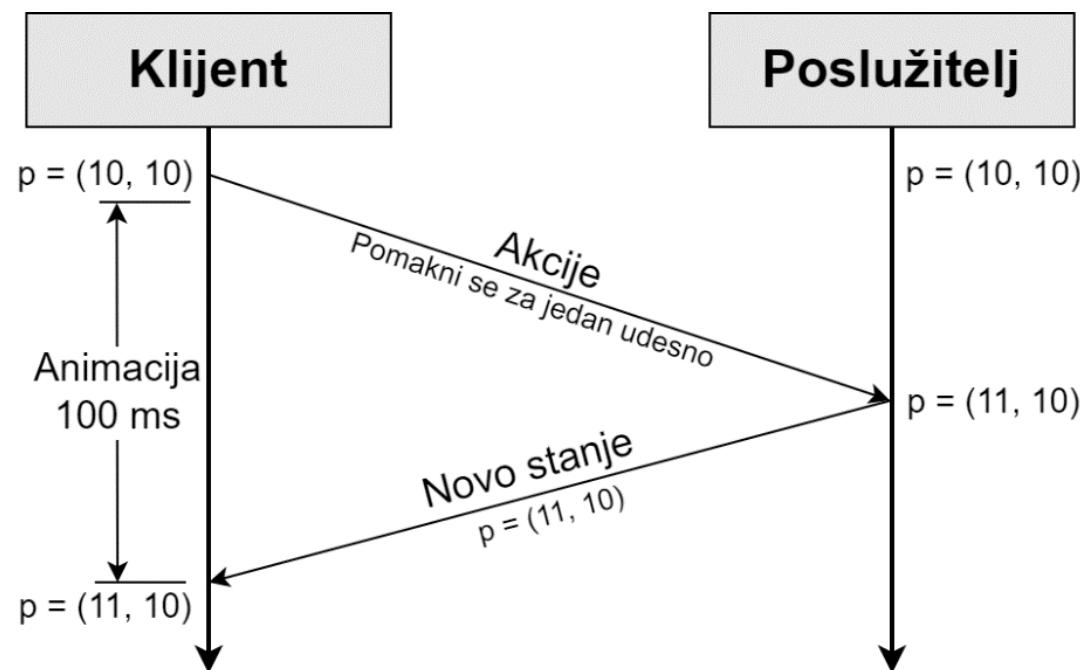
Interpolacija i ekstrapolacija

- **Interpolacija** označava proces u kojem se na temelju dvije poznate vrijednosti određene varijable izračunavaju među vrijednosti, odnosno omogućuje postupan prijelaz između tih vrijednosti,
- **Ekstrapolacija** označava proces procjene buduće vrijednosti određene varijable na temelju prethodnih vrijednosti.



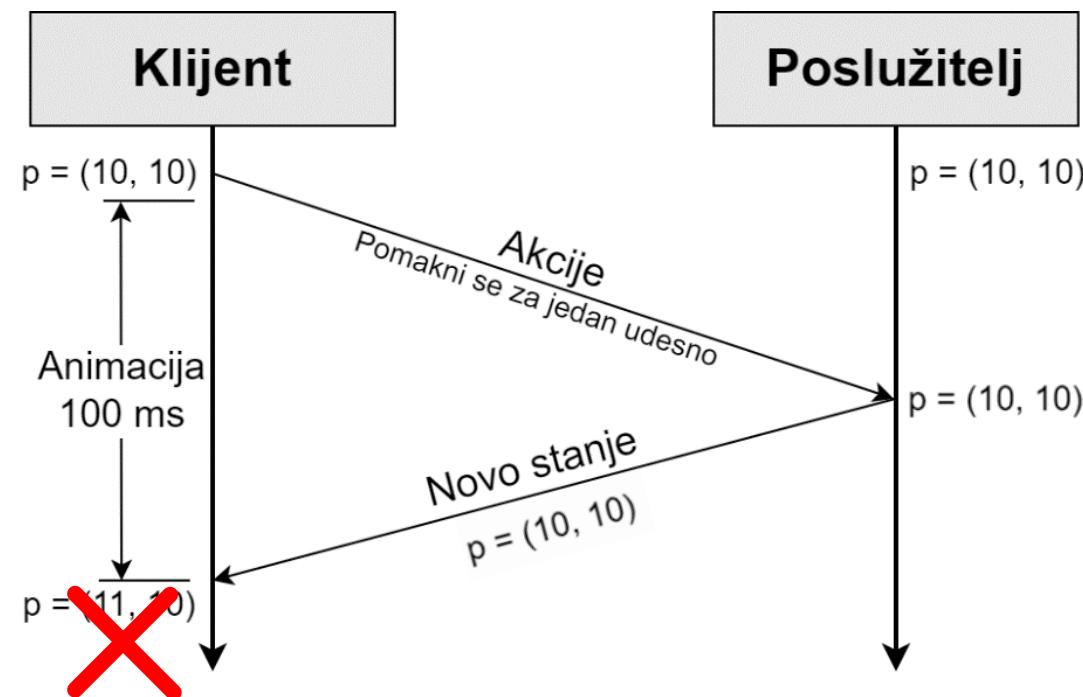
Predikcija ponašanja igrača

- Predikcija ponašanja igrača je tehnika u kojoj klijent kada napravi **komandu** ona se **izvršava odmah na njemu** odnosno izvršava se odgovarajuća animacija i izračun stanja na temelju komande, a ne čeka se potvrda od poslužitelja
- Vrlo efikasna metoda ako poslužitelj potvrdi naredbu izvršeno na klijentu



Što ako poslužitelj ne autorizira predviđenu poziciju?

- Kada poslužitelj ne autorizira predviđenu poziciju dolazi do razlike između stanja između klijenta i poslužitelja



Što ako predikcija nije u redu? Izmirenje!

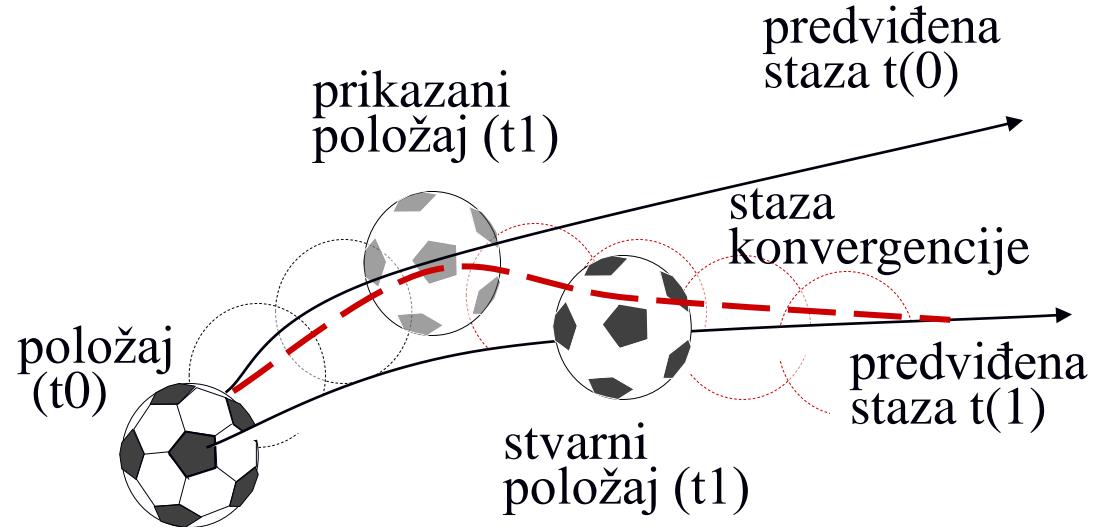
- Ako klijent napravi predikciju stanja, a koja nije u skladu s novi izračunatim stanjem na poslužitelju dolazi do razlike u stanju koja se mora izmiriti
 - Primjerice klijent je rekao da se pomjera ulijevo, ali neka efekt od drugog igrača ili kompjuterski kontroliranog lika na poslužitelju ga gurne u desno ili mu zaprijeći kretanje u lijevo
- **Izmirivanje stanja** je proces u kojem stanje na klijentu mora biti dovedeno u stvarno stanje na temelju osvježenja dobivenog od poslužitelja
 - **Postupno izmirivanje (konvergencija)** (engl. rubber banding) – je pristup u kome se poslužiteljska i klijentska simulacija je da se postupno dovedu u sinkronizirano stanje
 - **Instantno izmirivanje** (engl. snapping, teleporting) – je pristup u kojem se u idućem okviru klijentska simulacija odmah prilagodi poslužiteljskoj
- Postupno izmirivanje se provodi tako da se izračuna na temelju sadašnjeg poznatog stanja nekakvo buduće stanje te da se radi konvergencija iz sadašnjeg predviđenog stanja u buduće stanje kroz određenu krivulju
- U biti ta konvergencija je napravljena kroz proces ekstrapolacije (predvidimo gdje ćemo u budućnosti biti na temelju nove informacije) i interpolacije (interpoliramo međustanja koja su nam potrebna da dodemo do nove predviđene pozicije iz trenutne pogrešno predviđene pozicije)

Ekstrapolacija

- Svaki klijent na temelju osvježenja sa poslužitelja prikazuje pozicije i aktivnosti entiteta koje kontroliraju poslužitelj ili drugi klijenti
- U slučaju da osvježenja prestanu dolaziti zbog primjerice gubitka paketa u mreži može se izvršiti ekstrapolacija kako ti entiteti ne bi jednostavno stali
- Najpoznatiji algoritam za ekstrapolaciju je mrtva procjena (engl. Dead Reckoning)
- Mrtva procjena je proces izračunavanja trenutnog položaja nekog pokretnog objekta korištenjem prethodno utvrđenog položaja te vektora brzine i usmjerenja odnosno kinetičkih svojstava virtualnog objekta
- Korištenjem mrtve procjene mogu se entiteti nastaviti kretati u smjeru u kojem su se i kretali prije nego što su njihova osvježenja prestala dolaziti
- Nakon što osvježenja ponovno počnu dolaziti, slično kao kod predikcije ponašanja igrača treba izmiriti stanje na klijentu koje je izračunato temeljem mrtve procjene i stvarnog stanja na poslužitelju
- Jeste li nekada doživjeli da u umreženoj igri, često zbog degradacije u mrežnoj poveznici dođe do neresponsivnosti virtualnog svijeta, a svi likovi drugih igrača igri samo nastave pravocrtno trčati u smjeru u kojem su se kretali? To je Mrtva Procjena!

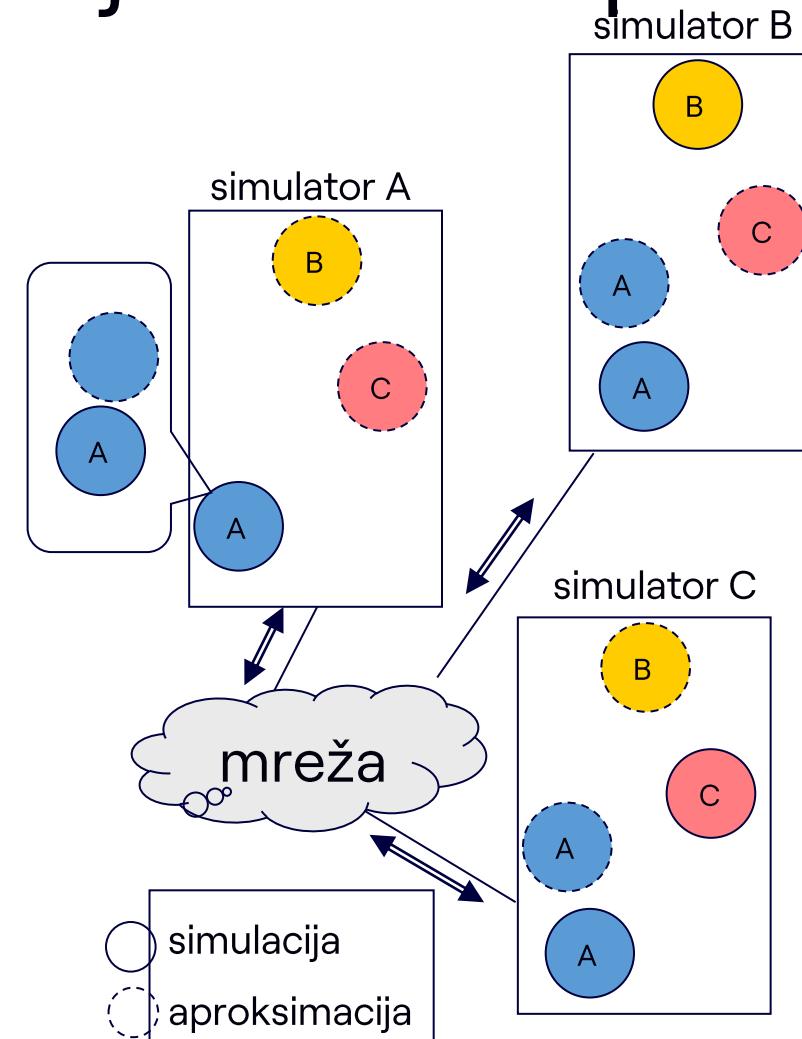
Ekstrapolacija – Mrtva procjena

- Sastoji se od dva osnovna dijela:
 - *Predikcija*: izračunavanje sadašnjeg stanja na temelju prethodno primljenih poruka osvježavanja
 - *Konvergencija (izmirenje)*: korekcija staze dobivene predikcijom na temelju novo-primljenih poruka osvježavanja (“izglađivanje”)
- Koristi se za entitete koje kontrolira udaljeni sudionik u simulaciji (primjerice poslužitelj)
- Izmirenje ili konvergencija se koristi kod bilo kakvog predviđanja bila to predikcija igrača ili mrtva procjena



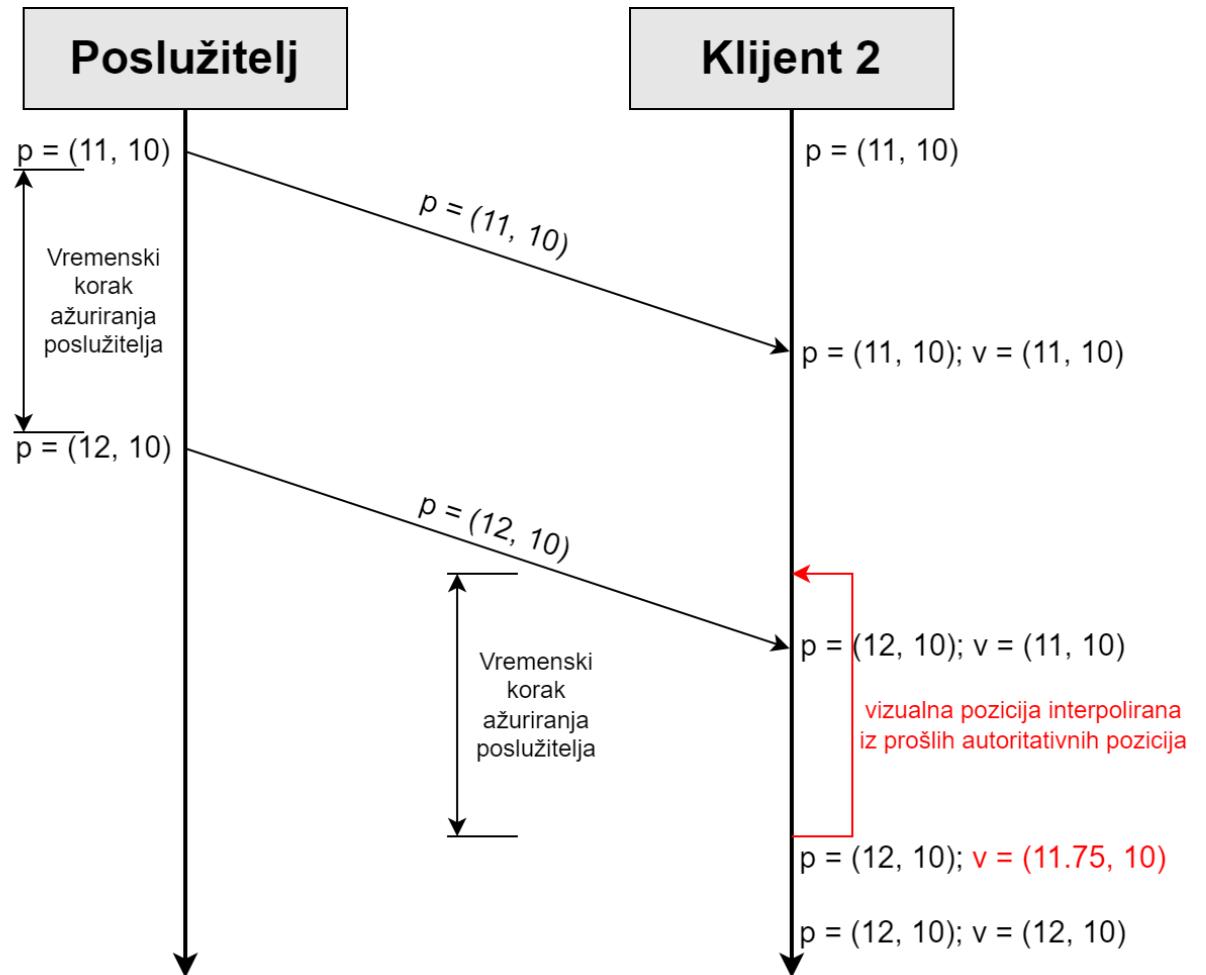
Mrtva procjena za smanjenje količine prometa

- Na strani svakog čvora koristi se i simulacija i predikcija trenutnog stanja na temelju posljednjeg poslanog osvježenja
- Predikcija je jednostavnija za računanje
- Svaki čvor za vlastite entitete šalje poruke osvježavanja samo ako razlika između simulacije i predikcije prijeđe zadani prag
- Svaki čvor radi predikciju (predviđanje) stanja udaljenih entiteta, npr. položaja i orientacije, na temelju lokalno pohranjene informacije
- Uvjet za primjenu metode je (barem djelomična) predvidivost kretanja entiteta kako bi se mogla primijeniti predikcija



Interpolacija entiteta

- Interpolacija entiteta je konceptualno vrlo slična prije objašnjenoj tehnici interpolacije snimaka, samo se u ovom slučaju radi o pojedinačnim entitetima u videoigri
- Interpolacija entiteta se radi kada imamo videoigre u kojima se mrtva procjena teško može primijeniti, primjerice kada se igrači mogu u trenutku zaustaviti se, promijeniti smjer i slično
- Kod interpolacije entiteta se ostali igrači prikazuju u prošlosti u odnosu na igrača korisnika
- Zatim se pozicije danih entiteta prikazuju na klijentskom računalu tako da se interpoliraju između dvije dobivene informacije odnosno osvježenja



Zadatak za iduće predavanje

- Pročitati članak i pogledati kratki video o tome kako radi proces stvaranja mečeva: <https://netduma.com/blog/how-matchmaking-works/>