

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

SEMINAR

**Primjena proširene stvarnosti za
edukaciju o svojstvima umreženih
videoigara: Analiza
konzistentnosti i razlike
uzrokovane kašnjenjem među
korisnicima mreže**

Dominik Arij

Voditelj: *Mirko Sužnjević*

Zagreb, svibanj 2023.

SADRŽAJ

1. Uvod	1
2. Svojstva umreženih videoigara	3
2.1. Slaba konzistentnost	4
2.2. Jaka konzistentnost	5
2.3. Kašnjenje	7
3. Metodologija implementacije	9
3.1. Osmišljanje i animacija modela	9
3.1.1. Model slabe i jake konzistentnosti	11
3.1.2. Model slučaja "smrti iza zida"	14
3.2. Razvoj sustava za označavanje dijelova modela	17
4. Rezultati i rasprava	21
5. Zaključak	24
6. Literatura	25
7. Sažetak	27

1. Uvod

Umrežene videoigre (engl. *multiplayer games*) postale su jednom od najpopулarnijih formi zabave okupljuјući milijune korisnika diljem svijeta. Moguće ih je igrati na računalima, konzolama, mobilnim uređajima i drugim platformama koje omogуćuju interakciju među igračima. Za razliku od tradicionalnih igara, umrežene videoigre igračima omogуćuju međusobnu komunikaciju, timski rad i gradnju strategije kako bi se ostvario krajnji cilj. Kreatori takvih igara stvaraju svjetove koji su trajni te omogуćuju uspostavljanje društvenih odnosa, kako bi se poboljšale performanse korisnika [1].

Za dojam kvalitete iskustva korisnika (engl. *Quality Of Experience - QoE*) pri-likom igranja, zaslužno je stanje mreže opisano njenim svojstvima. Jedno od takvih svojstava je **konzistentnost** (engl. *network consistency*), koja se odnosi na sposobnost mreže da održava isto stanje igre na svim uređajima igrača u stvarnom vremenu. Koncept **slabe konzistentnosti** (engl. *eventual consistency*) u igrama, odnosi se na situaciju u kojoj se stanja igre ne podudaraju u realnom vremenu na svim korisničkim uređajima, dok koncept **jake konzistentnosti** (engl. *strong consistency*) implicira da se stanja igre uvijek podudaraju. Jedan od ključnih elemenata za postizanje zadovoljstva igrača tijekom igranja jest osigurati im to da će njihove akcije biti učinjene u stvarnom vremenu te da će se stanje igre jednakо prikazivati na svim udaljenim uređajima. U stvarnosti se potpuna konzistentnost smatra neostvarivom [2].

Kašnjenje (engl. *latency*) je još jedna od važnih karakteristika višekorisničkih mreža. Ono se odnosi na vremensku razliku između trenutka kada igrač izvrši neku akciju te trenutka kada se ta akcija prikaže na ekranu ostalih igrača. Ako je kašnjenje veliko, to može negativno utjecati na kvalitetu iskustva igrača uzrokujući frustraciju i nelagodu. Ako vremensko kašnjenje u igri premaši njezin unaprijed utvrđeni prag, igra postaje nestabilnom pa je time smanjenje kašnjenja jedan od glavnih izazova u razvoju umreženih videoigara [3].



Slika 1.1: Simbolična usporedba normalnih mrežnih uvjeta i mreže s visokom kašnjenjem.
Slika preuzeta iz [4].

Koristeći prepostavke o spomenutim svojstvima mreže kod umreženih videoigara, u ovom će radu biti opisana implementacija animiranih 3D modela unutar prethodno razvijene edukativne mobilne aplikacije. Za vrijeme pisanja ovog rada aplikacija se nalazila u fazi testiranja i dorade, a razvijena je kao pokazni materijal diplomskog rada studentice Ane Sundji na Fakultetu Elektrotehnike i Računarstva u Zagrebu, 2022. godine [5]. Modeli navedenih svojstava se moraju kontinuirano prikazivati u proširenoj stvarnosti (engl. *Augmented Reality - AR*) sa svojim funkcionalnostima (skaliranje i rotacija) te moraju tematski odgovarati sadržaju iz poglavlja *Umrežene videoigre* trenutno nedovršenog udžbenika za srednje škole imena *Razvoj višekorisničkih igara*.

2. Svojstva umreženih videoigara

Kako bismo mogli izraditi 3D modele koji vjerno prikazuju željena svojstva mreže, potrebno je razumjeti način na koji se informacije prenose između računala u mreži te kako se lokalna stanja korisničke sjednice ažuriraju nakon što je prema mreži upućen zahtjev za promjenom globalnog stanja igre. Upravljanje vremenom ključno je za izvršavanje zadatka unutar vremenskih ograničenja sustava, a poruke moraju biti prenesene, među entitetima koji komuniciraju, u pravodobnom roku [6]. U nastavku poglavlja detaljnije ćemo se osvrnuti na svojstva slabe i jake konzistentnosti te na kašnjenje mreže u kontekstu umreženih videoigara.

PROIZVODNJA KOMPETITIVNIH VIDEOIGARA

3 Umrežene videoigre

Nakon ovog poglavlja moći ćete:

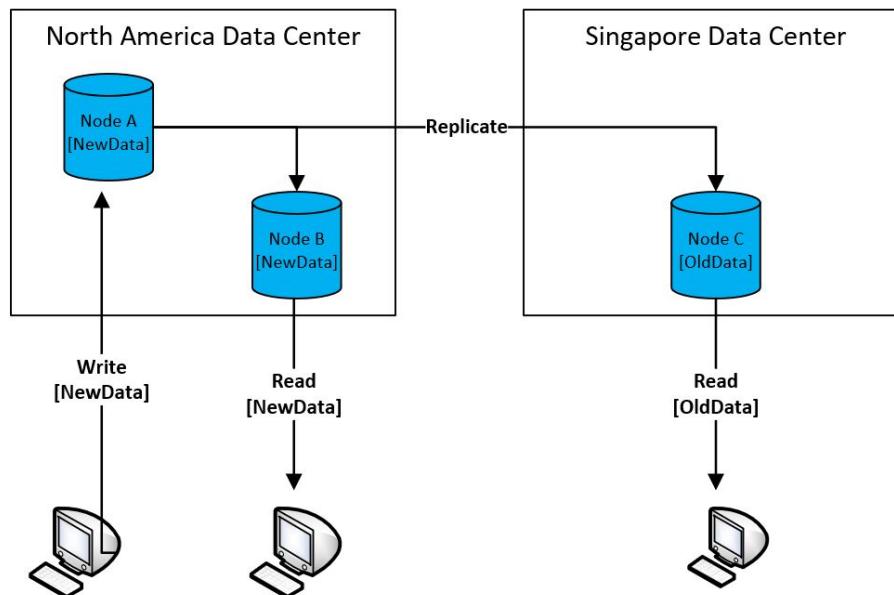
- objasniti osnovne koncepte umrežene simulacije,
- objasniti koncept konzistentnosti te dvaju osnovnih pristupa održavanju konzistentnosti u umreženim igrama,
- objasniti osnovne metode za implementaciju umreženih igara (priključnice, serijalizaciju i poziv udaljene procedure),
- objasniti slijed funkcija koji je u umreženoj igri potrebno izvršiti da bi se igraču prikazao rezultat komandi koje je dao igri,
- nabrojati karakteristike mreže koje utječu na iskustvenu kvalitetu videoigara i
- objasniti metode za sakrivanje kašnjenja kod umreženih videoigra.

Slika 2.1: Odjeljak iz spomenutog udžbenika koji prikazuje ishode učenja poglavlja vezanog uz umrežene igre.

2.1. Slaba konzistentnost

Slaba konzistentnost u kontekstu umreženih videoigara ne garantira da će se redoslijed i vidljivost ažuriranja na svim čvorovima (računalima) uvijek podudarati [7]. Čim se zahtjev za ažuriranjem pošalje bilo kojem čvoru, sustav izvršava odgovarajuću operaciju i taj se čvor prvi ažurira [7]. Naknadno, te se informacije prenose na sve ostale čvorove što znači da, ako se na čvoru koji još nije ažuriran izvrši operacija čitanja, taj čvor može vratiti zastarjele informacije [7]. Konačno, slaba konzistentnost garantira da će poruke biti dostavljene svim čvorovima ali ne garantira kada će se to dogoditi [8].

Ovakav problem negativno utječe na iskustvo igrača jer igrači mogu biti sprječeni u izvođenju svojih akcija ili su prisiljeni poništiti ih zbog neslaganja među klijentima. To može dovesti do frustracije i gubitka vremena. Tehnike poput **replikacije podataka** koriste se u umreženim videoigramama za rješavanje problema slabe konzistentnosti. Replikacijom se stvaraju kopije podataka na različitim serverima kako bi se osigurala konzistentnost. Algoritmi za upravljanje konzistentnošću, uključujući one zasnovane na vektorskem satu i preslikavanju stanja, također se pokazuju učinkovitim. Slika 2.2 prikazuje sustav u kojem neki od čvorova dostupnih za čitanje sadržaja, pod utjecajem slabe konzistentnosti, čitaju zastarjele podatke.

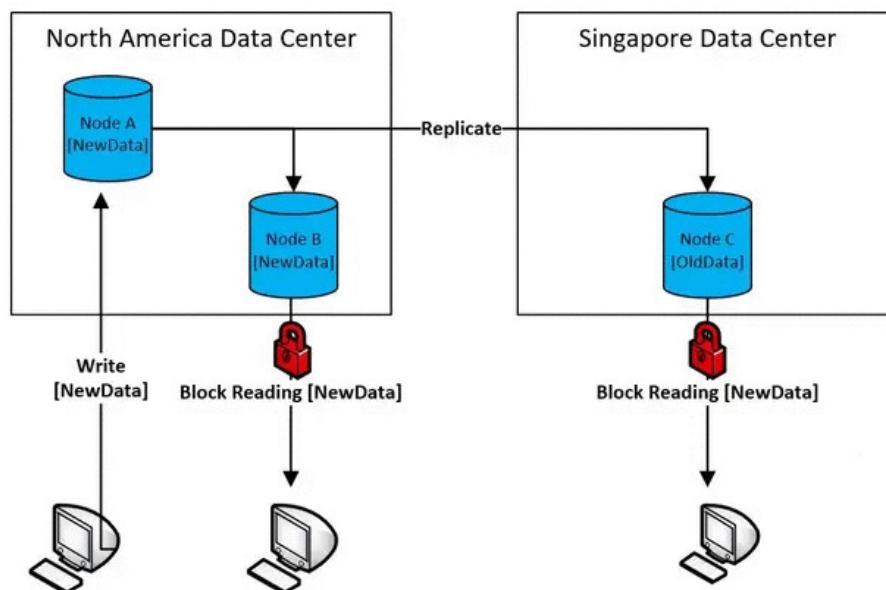


Slika 2.2: Čitanje zastarjelih podataka pod utjecajem slabe konzistentnosti unutar raspodijeljenog sustava. Slika preuzeta iz [9].

2.2. Jaka konzistentnost

Kod jake konzistentnosti, promjene stanja igre moraju biti vidljive svim igračima u isto vrijeme, bez obzira na njihovu lokaciju ili uređaj koji koriste. Nakon što se pošalje zahtjev za ažuriranje nekom čvoru, svi ostali čvorovi se koordiniraju kako bi se osigurala sinkronizacija operacije [7]. Tijekom vremena potrebnog za ažuriranje svih čvorova novim zahtjevom za ažuriranje, odgovor na bilo koji sljedeći zahtjev za ažuriranje od bilo kojeg čvora bit će odgođen jer svi čvorovi komuniciraju međusobno kako bi se postigla konzistentnost [7]. Čim se svi entiteti usklade, počinje obrada sljedećih zahtjeva za ažuriranje [7].

Jaka konzistentnost pruža ispravnost, ali pod cijenu često visokih kašnjenja [10]. Postizanje jake konzistentnosti u umreženim videoigrama ključno je za ostvarivanje dosljednog i ugodnog iskustva igračima. Osim toga, jaka konzistentnost može smanjiti potrebu za nadzorom i regulacijom igara, jer će igrači imati jednake uvjete igranja igre bez obzira na to gdje se nalaze ili koji uređaj koriste. Slika 2.3 prikazuje sustav u kojem svi čvorovi dostupnih za čitanje sadržaja, pod utjecajem jake konzistentnosti, čitaju najnovije i ažurirane podatke.



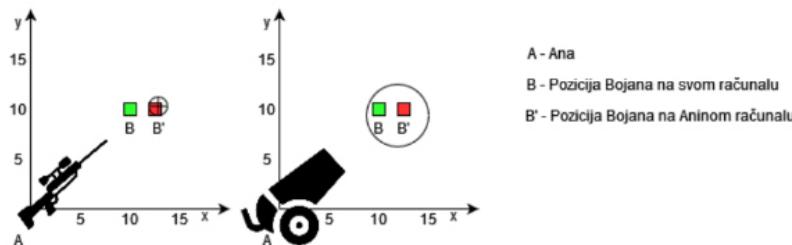
Slika 2.3: Čitanje ažuriranih podataka pod utjecajem jake konzistentnosti unutar raspodijeljenog sustava. Slika preuzeta iz [9].

Unutar Tablice 2.1 prikazana je usporedba slabe i jake konzistentnosti prema cilju, pouzdanosti, implementaciji, performansama i kašnjenju te dostupnosti. Ova je tablica izvadak iz usporedbe prikazane u [7].

Kriterij	Jaka konzistentnost	Slaba konzistentnost
Cilj	Pružanje ispravnosti	Pružanje boljih performansi
Redoslijed i vidljivost ažuriranja	Isti na svim čvorovima	Nisu isti na svim čvorovima
Implementacija	Složenije i skuplje	Jednostavnije i jeftinije
Performanse i kašnjenje	Slabije performanse i veće kašnjenje	Bolje performanse i manje kašnjenje
Dostupnost	Niža dostupnost	Viša dostupnost

Tablica 2.1: Usporedba karakteristika slabe i jake konzistentnosti prema [7].

Slika 55 ilustrira kako brojčano ista nekonzistentnost može uzrokovati različite efekte i dojmove kod igrača. Kad Ana gada snajperom neće pogoditi Bojana jer gađa u poziciju kojom oružje neće zahvatiti Bojana te time je Anin dojam utjecaja nekonzistentnosti velik. Kada Ana gada topom Bojana pogodit će ga i tada će Anin dojam utjecaja nekonzistentnosti biti zanemariv.



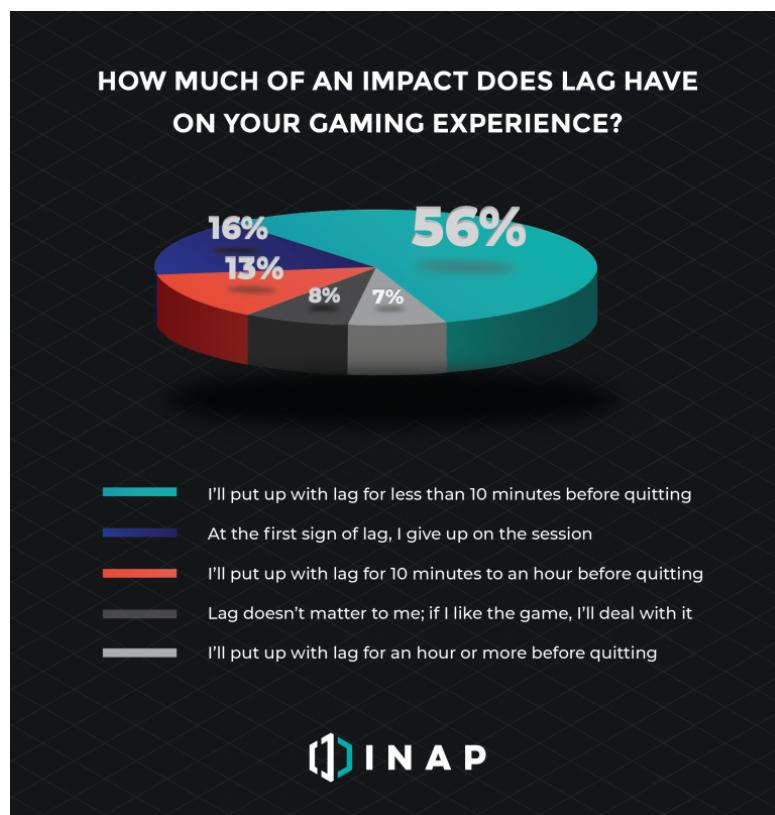
Slika 55 Razlika u konzistentnosti i utjecaj na dojam u igri

Slika 2.4: Odjeljak iz spomenutog udžbenika koji prikazuje utjecaj nekonzistentnosti na kvalitetu videoigre.

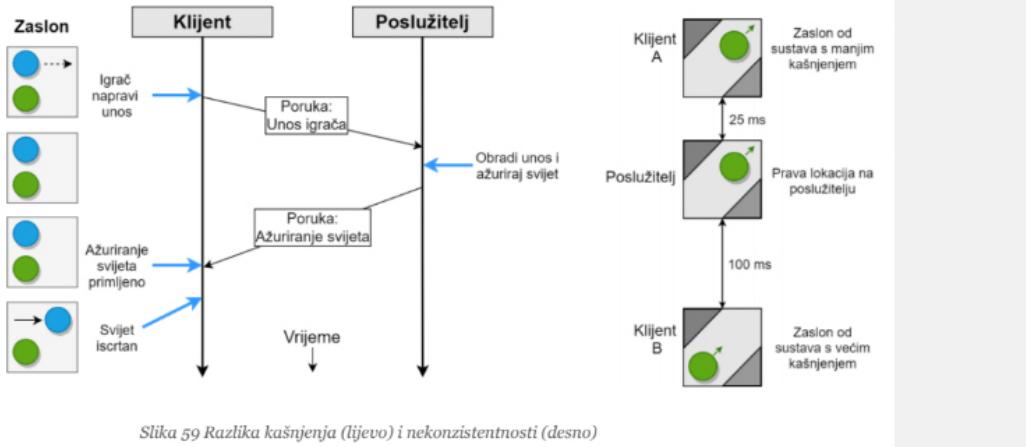
2.3. Kašnjenje

Kod umreženih videoigara, uzrok kašnjenja može se nalaziti u raznim aspektima mreže i njene infrastrukture. Jedan od najčešćih uzroka kašnjenja je vrijeme potrebno za prijenos podataka zbog udaljenosti između igrača i servera (engl. *host*) na kojem se igra odvija. Takva se vrsta kašnjenja naziva *ping*. Osim toga, kašnjenje može nastati zbog preopterećenja mreže, loše kvalitete veze između igrača i servera, tehničkih problema na serveru ili igračevom računalu te drugih čimbenika.

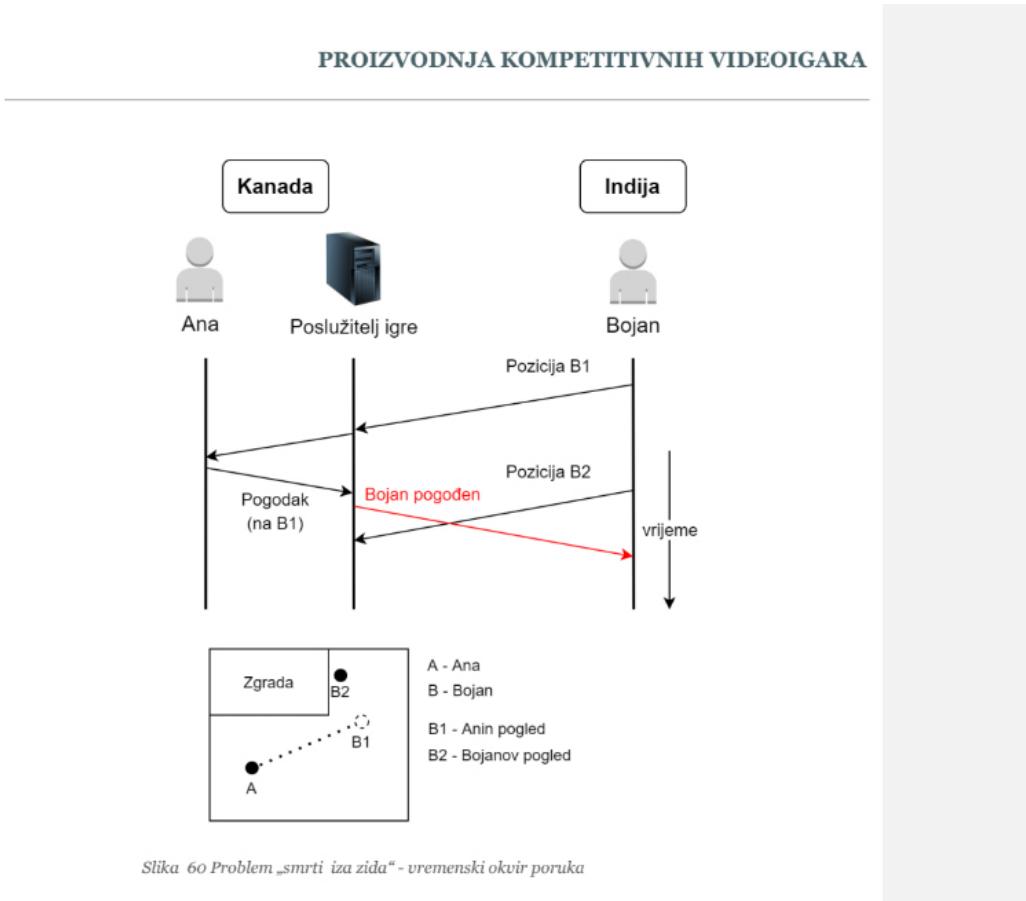
Ovakav problem, može uzrokovati da igrači ne vide točno gdje se drugi igrač nalazi u stvarnom vremenu, što može dovesti do nelogičnih situacija. Primjerice, u igrama pucanja, igrači se često oslanjaju na brze reakcije i precizno ciljanje kako bi ostvarili prednost nad svojim protivnicima. Takve su akcije znatno otežane kada je u sustavu prisutno kašnjenje, a ono može narušiti i doživljaj igrača te umanjiti kvalitetu igre. Slika 2.5 prikazuje u kolikoj mjeri i na koji način kašnjenje negativno utječe na iskustvo igrača prema rezultatima iz [11].



Slika 2.5: Reakcije igrača na prisustvo kašnjenja za vrijeme igranja videoigara. Slika preuzeta iz [11].



Slika 2.6: Slika iz spomenutog udžbenika koja prikazuje utjecaj kašnjenja na konzistentnost videoigre.



Slika 2.7: Slika iz spomenutog udžbenika koja je korištena kao referenca za izradu animacije slučaja "smrti iza zida".

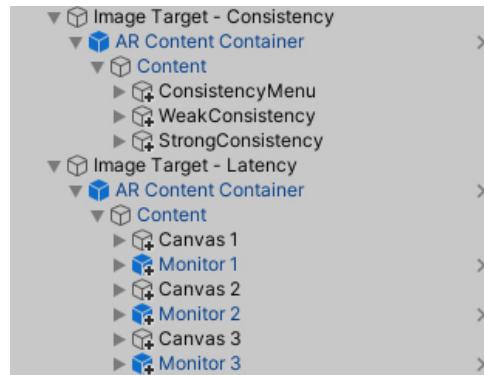
3. Metodologija implementacije

Kako bi bilo moguće doraditi postojeću verziju AR aplikacije pomoću modela koji prikazuju svojstva umreženih videoigara, kloniran je **Git** repozitorij aplikacije u kojem su na nekoliko grana bile sadržane različite funkcionalnosti. Izrađena je nova grana, a projekt je uređivan korištenjem pogona za razvoj igara **Unity** u inačici 2021.3.21f1 *Personal*. Dodatno, kako bi se omogućio mobilni razvoj za operacijski sustav Android, postavljena je konfiguracija projekta koja omogućuje stvaranje datoteke formata **APK** (engl. *Android Package Kit*) uz potrebne alate poput JDK (engl. *Java Development Kit*) te Android SDK (engl. *Software Development Kit*). Za uređivanje skripti u programskom jeziku C# korišteno je integrirano razvojno okruženje **Microsoft Visual Studio** u inačici *Community* 2022.

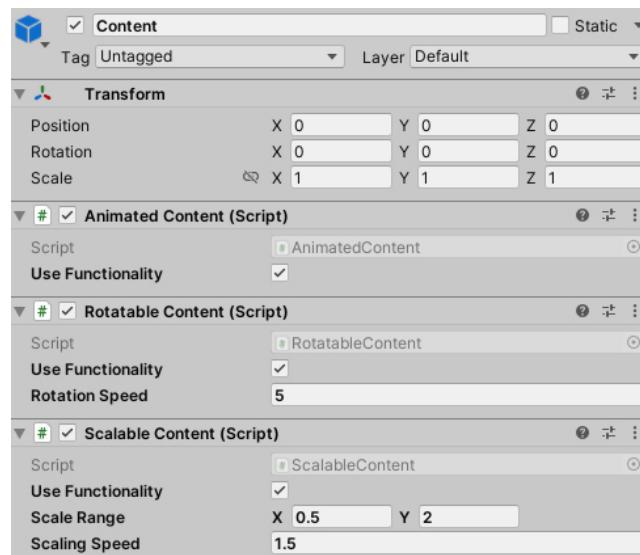
Prikaz sadržaja u proširenoj stvarnosti omogućen je korištenjem platforme **Vuforia Engine** u inačici 10.7.2., koja je prethodno dodana u izvorni projekt korištenjem ugrađenog upravitelja paketima. Kako bi se omogućilo prikazivanje novih AR sadržaja, u bazu slikovnih oznaka dodana je još po jedna za svaki od modela koji želimo prikazati.

3.1. Osmišljanje i animacija modela

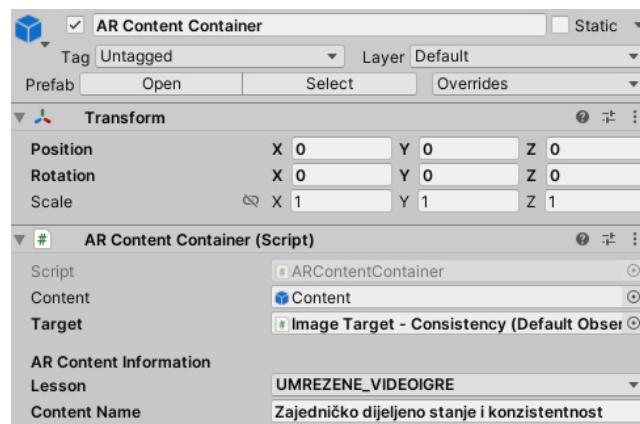
Konkretni 3D objekti s pripadajućim animacijama smješteni su unutar roditeljskog objekta *Content*, koji pomoću skripti *Rotatable Content* i *Scalable Content* omogućuje ispunjenje zahtjeva za skaliranjem i rotiranjem prikazanih modela. S obzirom na to da su ove skripte stvorene unutar izvorne verzije aplikacije, njihova struktura neće biti detaljnije objašnjena u okviru ovog rada. Također, izostavljamo detaljnije pojašnjenje roditeljskih objekata *AR Content Container* i *Image Target*, koji omogućuju prepoznavanje slikovne oznake za prikaz sadržaja unutar AR scene te postavljanje pripadajućeg teksta za model koji je prikazan. Hijerarhija navedenih objekata prikazana je na Slici 3.1. Slike 3.2 i 3.3 prikazuju primjer konfiguracije komponenti objekata *Content* i *AR Content Container* dok će dijelovi konfiguracije objekta *Image Target* biti prikazani u Odjeljku 3.2.



Slika 3.1: Hijerarhija objekata unutar kojih se nalaze modeli svojstava umreženih videoigara.



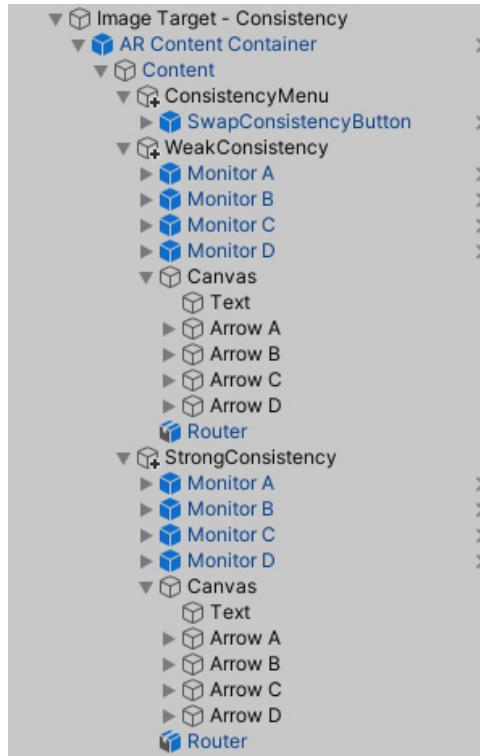
Slika 3.2: Konfiguracija komponenti objekta *Content*.



Slika 3.3: Primjer konfiguracije komponenti objekta *AR Content Container* za model konzistentnosti.

3.1.1. Model slabe i jake konzistentnosti

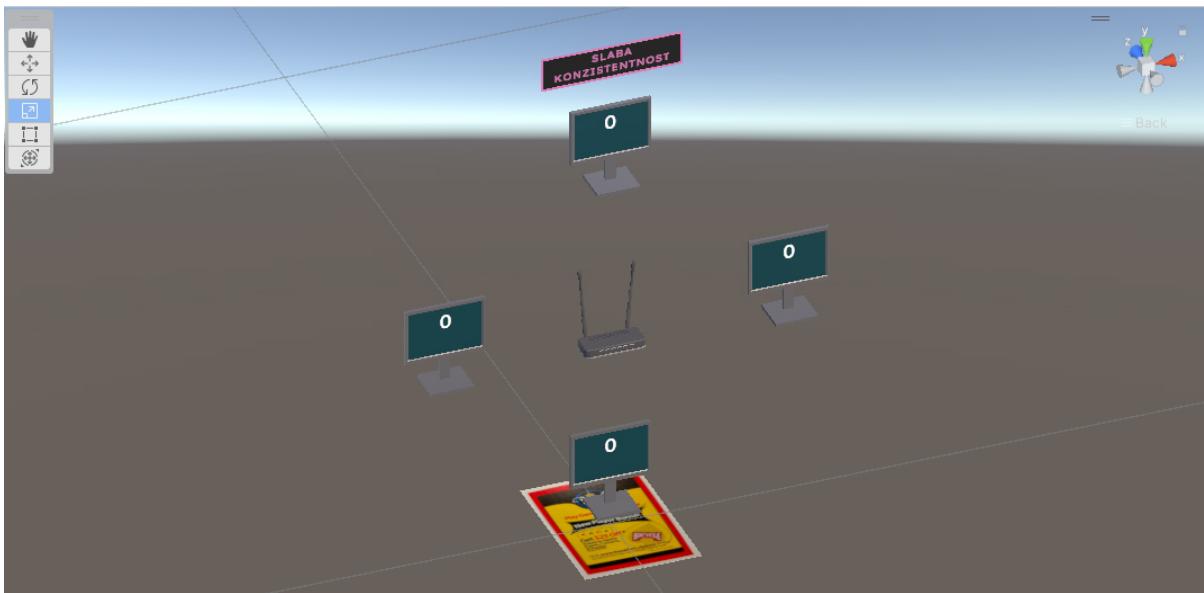
Slika 3.4 prikazuje proširenu objekta koji sadrži model konzistentnosti.



Slika 3.4: Proširena hijerarhija dijelova modela konzistentnosti.

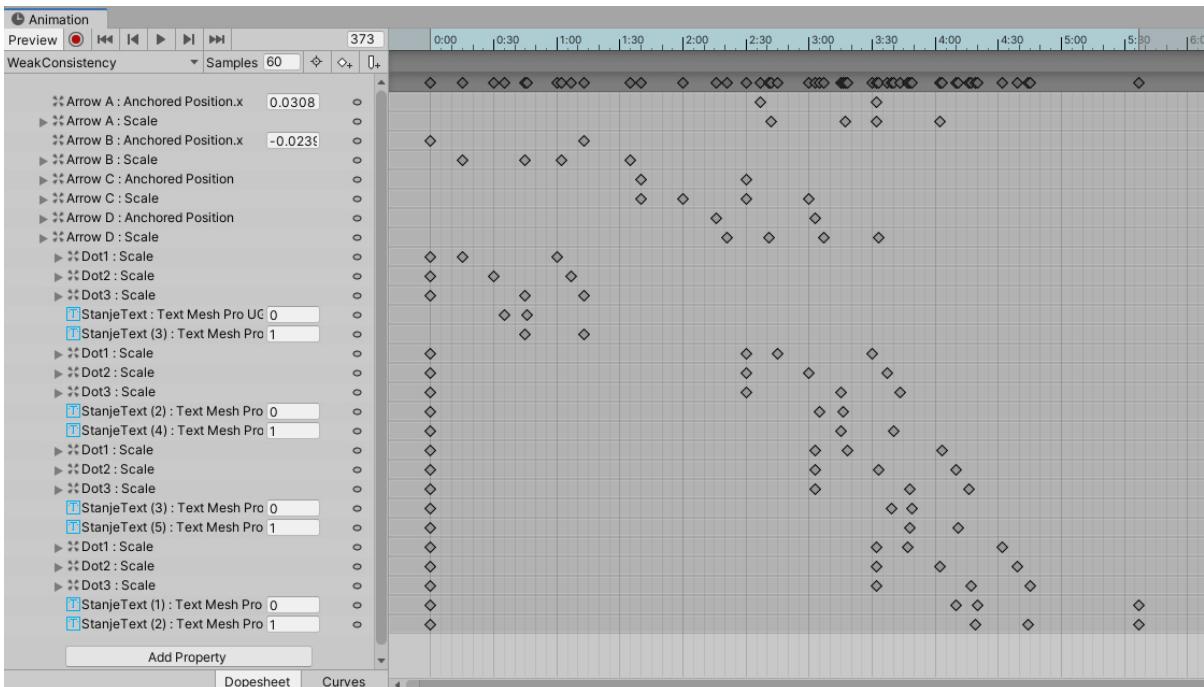
Kao što je moguće primijetiti, postoje dva različita prikaza konzistentnosti označenih objektima *Weak Consistency* i *Strong Consistency*. Gumb *Swap Consistency Button*, pomoću istoimene skripte, omogućuje da se pritiskom na njega uključi odnosno isključi pojedini prikaz te da se tekst gumba mijenja ovisno o trenutno aktivnoj vrsti konzistentnosti. Detalji skripte nisu predviđeni s obzirom na trivijalnost, a izgled gumba prikazan je na Slici 3.5 na samom vrhu modela.

Prikazi se sastoje od dijelova koji nisu animirani (3D modeli monitora i usmjeritelja preuzeti iz trgovine **Unity Asset Store**) i jednaki su za oba prikaza te različitim animiranim UI (engl. *User Interface*) objekata u obliku strelica za prijenos informacija, brojčane označe trenutnog stanja svakog računala te naznake za učitavanje stanja. Označe trenutnog stanja te naznake za učitavanje nalaze se unutar objekta *Canvas* svakog monitora. Kompozicija 3D objekata koji čine modele konzistentnosti, prikazana je na Slici 3.5.

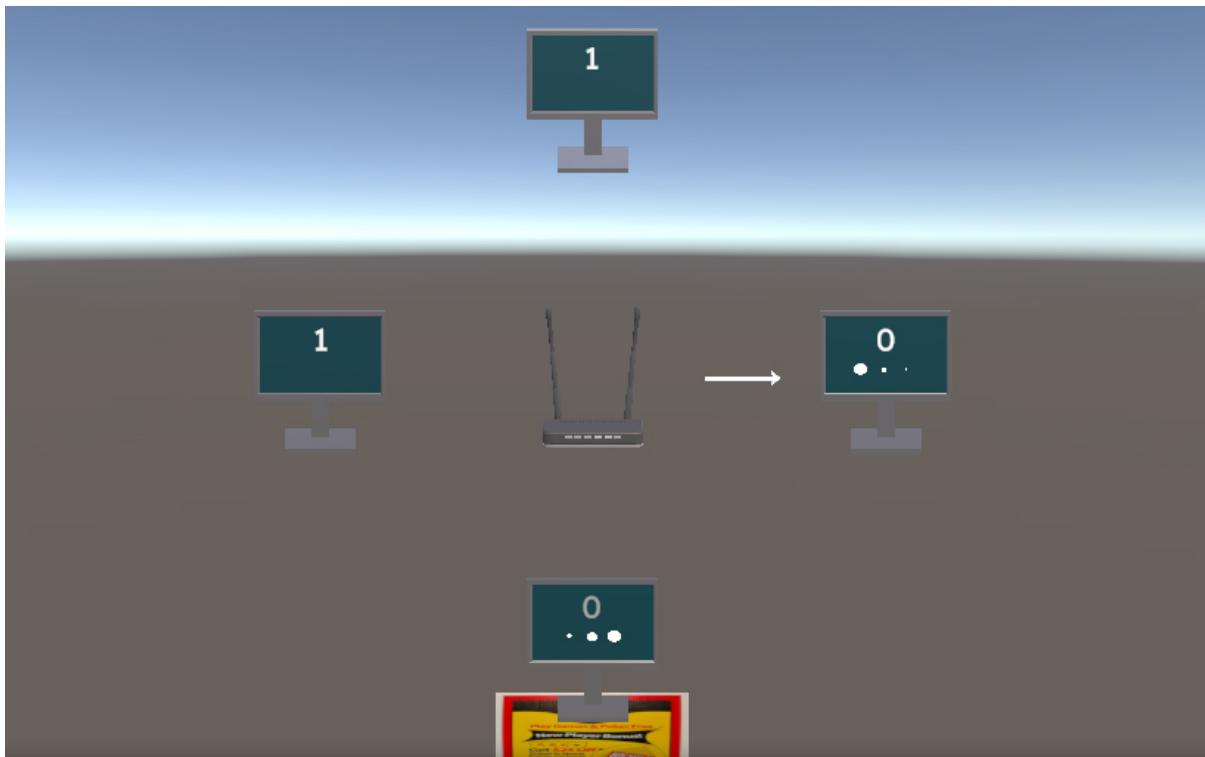


Slika 3.5: Kompozicija 3D objekata koji čine modele konzistentnosti.

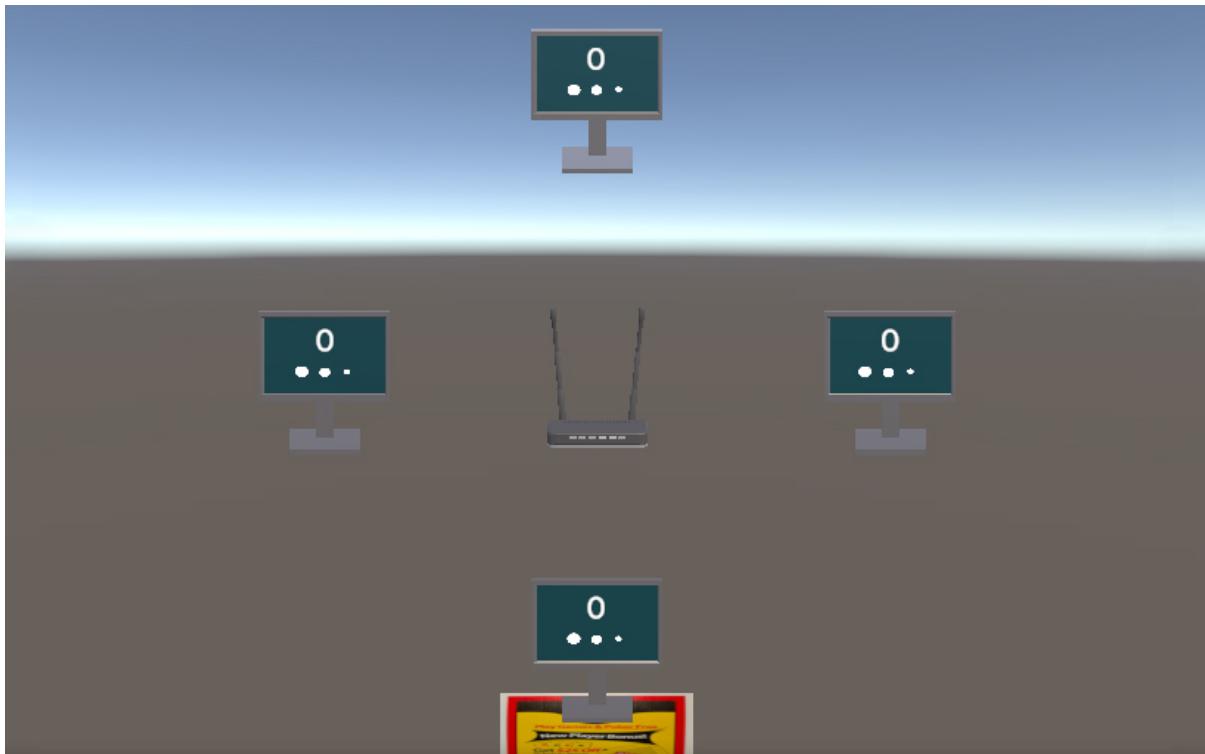
Koristeći komponentu *Animator*, za svaki od dva prikaza konzistentnosti, izrađena je složena animacija koja prikazuje način na koji se zahtjeva promjena stanja unutar mreže računala te naknadno ažuriranje stanja na računalima svih korisnika (Slika 3.6).



Slika 3.6: Struktura animacije slanja informacija i ažuriranja stanja kod prikaza slabe konzistentnosti.



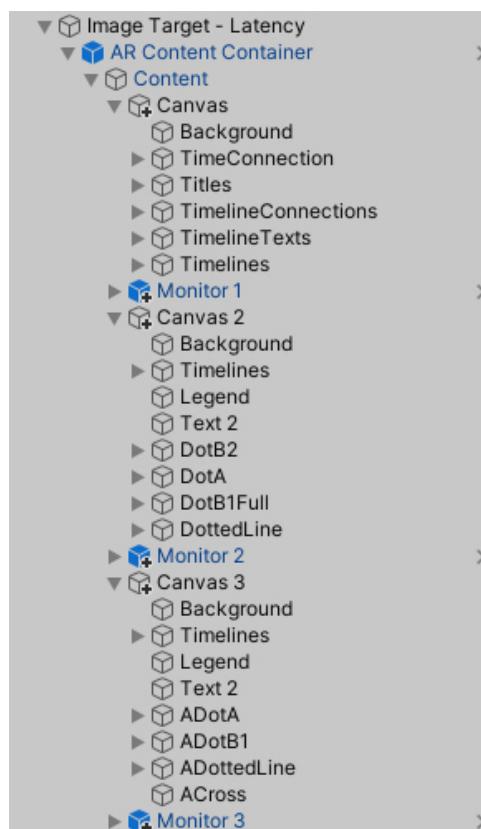
Slika 3.7: Značajan trenutak unutar animacije za prikaz slabe konzistentnosti.



Slika 3.8: Značajan trenutak unutar animacije za prikaz jake konzistentnosti.

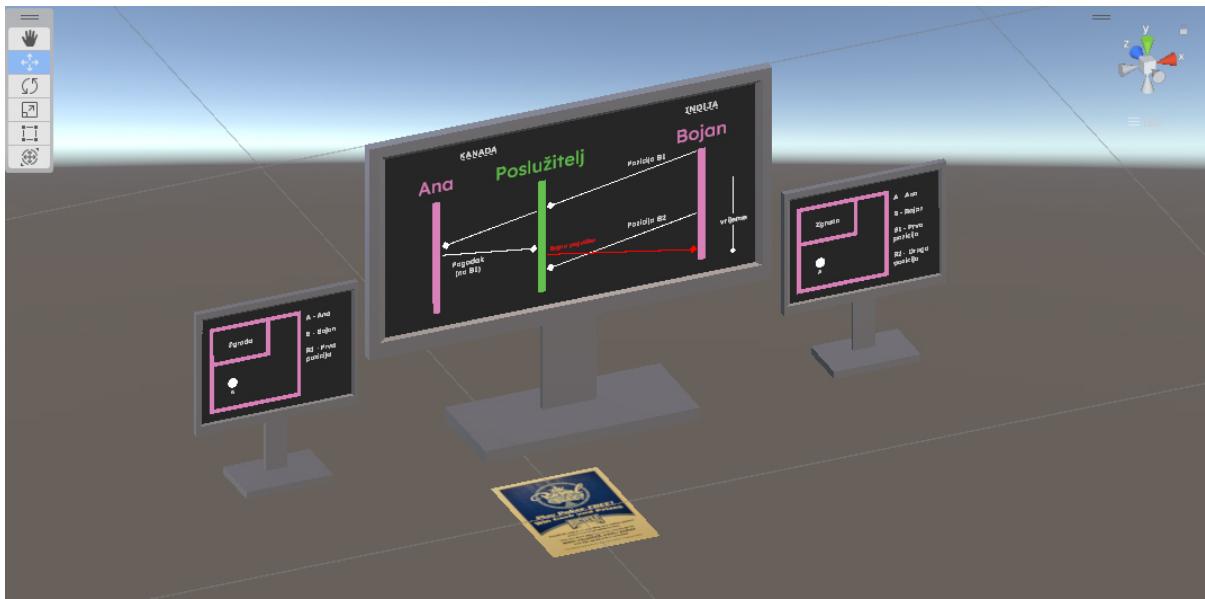
3.1.2. Model slučaja "smrti iza zida"

Kako bi predočili probleme koji nastaju uslijed mrežnog kašnjenja, kao model kašnjenja prikazali smo dijagram i lokalna stanja specifičnog slučaja "smrti iza zida". Dva igrača istovremeno igraju umreženu videoigru na zajedničkom poslužitelju. Prvi igrač ispaljuje metak prema drugom igraču koji pretrčava područje od izložene do sigurne pozicije. U trenutku pretrčavanja događa se kašnjenje te prvi igrač na svojem računalu vidi kako je pogodio drugog, a drugi igrač vidi kako je sigurno dotrčao do zaklona. Kao rezultat kašnjenja, lokalno stanje drugog igrača prikazuje smrt naknadno, nakon što je uspješno stigao do sigurne pozicije. To predstavlja suprotnost fizičkim načelima igre i uzrokuje nezadovoljstvo korisnika. Hijerarhija objekata koji čine prikaz opisanog slučaja unutar virtualne scene prikazana je na Slici 3.9.



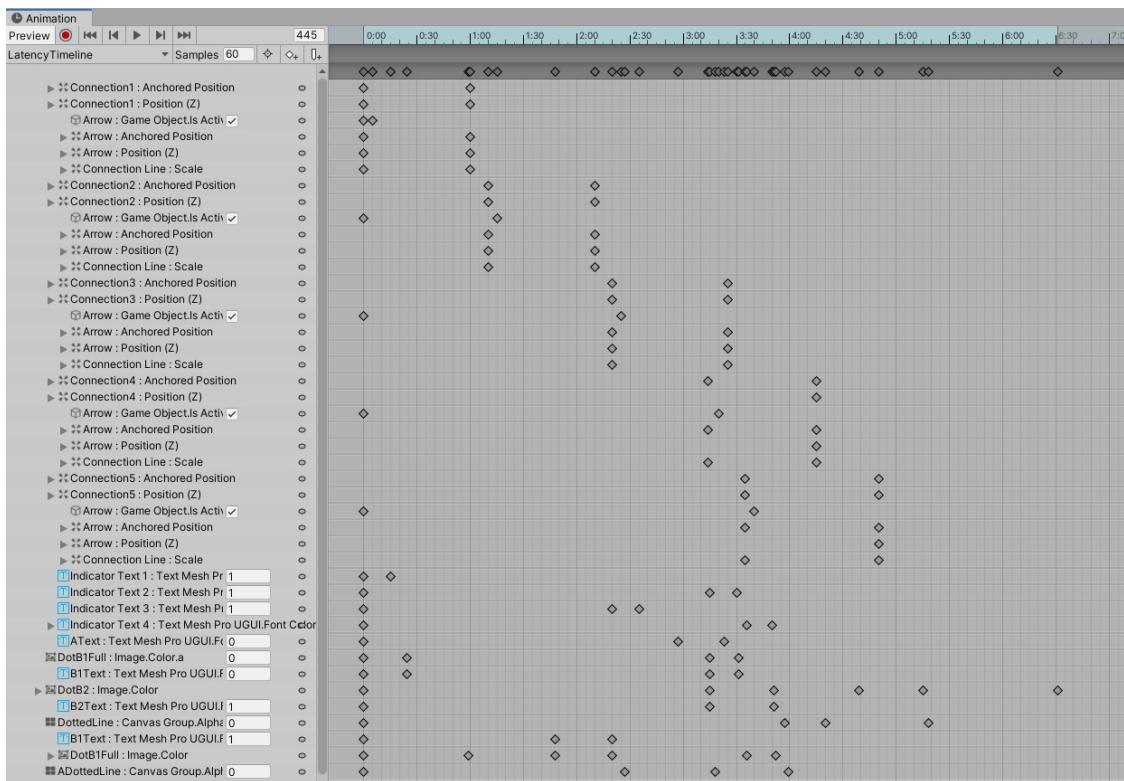
Slika 3.9: Proširena hijerarhija dijelova modela slučaja "smrti iza zida".

Na sličan način kao i kod prikaza konzistentnosti, ovaj se model sastoji od 3D objekata monitora od kojih jedan prikazuje dijagram opisanog slučaja, a preostala dva prikazuju lokalna stanja dvaju igrača u vremenu.



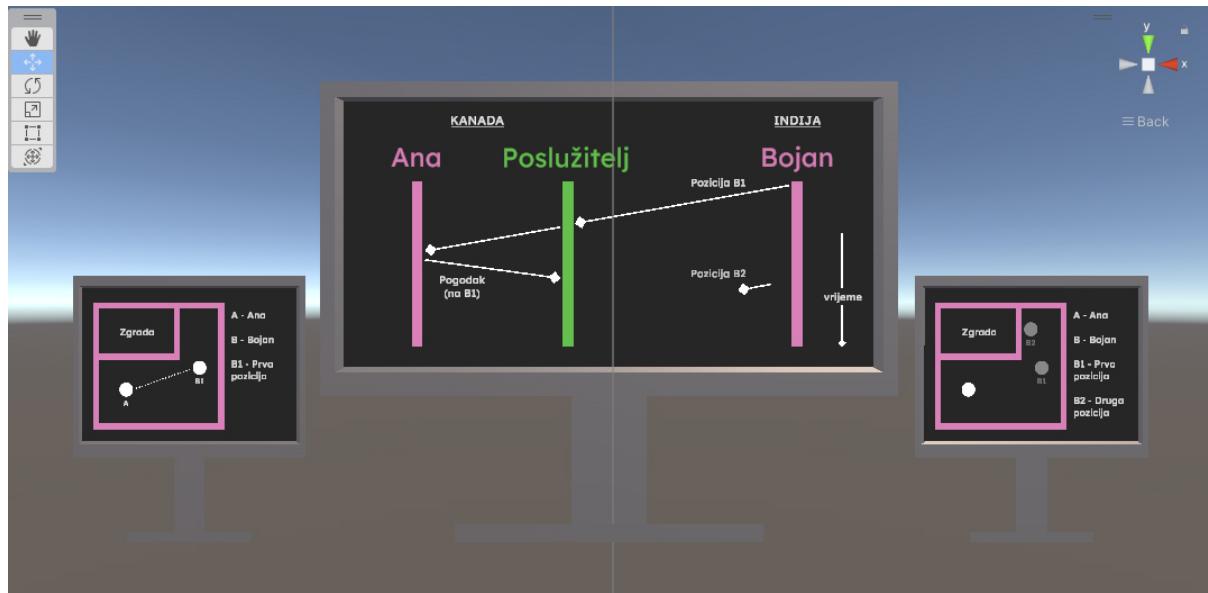
Slika 3.10: Kompozicija 3D objekata koji čine model slučaja "smrti iza zida".

Slika 3.11 prikazuje strukturu animacije koja ažurira stanje dijagrama i prikaze lokalnih stanja mijenjajući velicine, položaje, boje i vidljivost pojedinih elemenata objekta *Canvas* u vremenu.

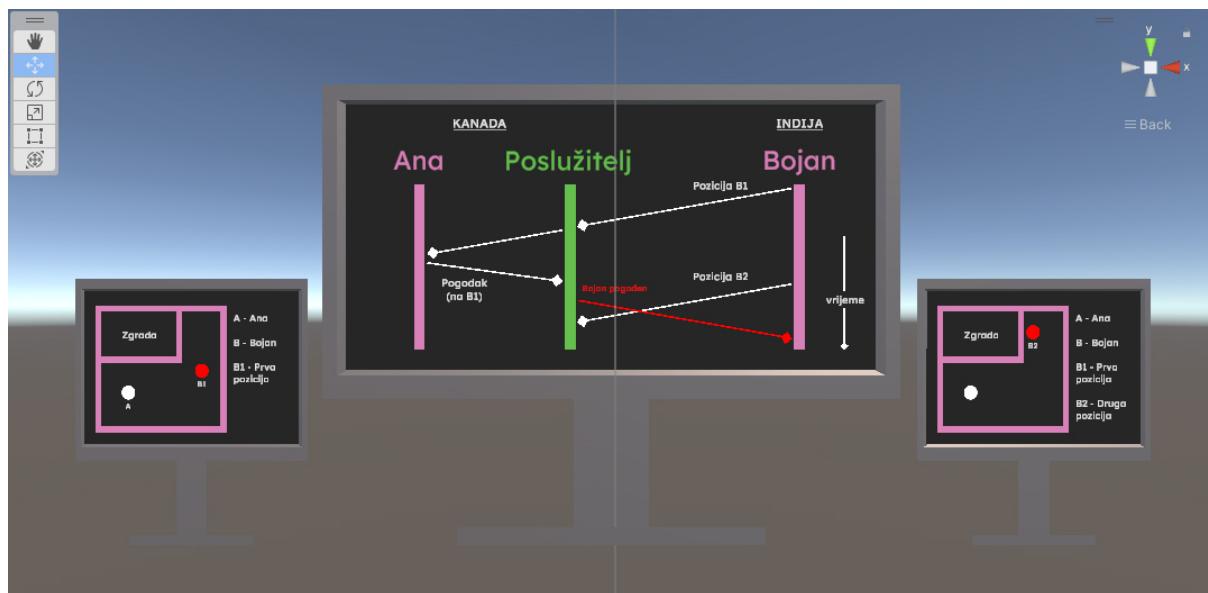


Slika 3.11: Struktura animacije dijagrama i promjene lokalnih stanja kod prikaza slučaja "smrti iza zida".

Na slikama 3.12 i 3.13 prikazana su dva značajna trenutka unutar animacije za prikaz slučaja "smrti iza zida". Središnji monitor prikazuje globalni razvoj događaja u vremenu dok manji monitori prikazuju lokalna stanja pojedinih igrača na koje kašnjenje utječe.



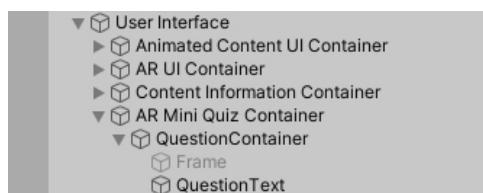
Slika 3.12: Značajan trenutak unutar animacije za prikaz slučaja "smrti iza zida".



Slika 3.13: Krajnje stanje animacije za prikaz slučaja "smrti iza zida".

3.2. Razvoj sustava za označavanje dijelova modela

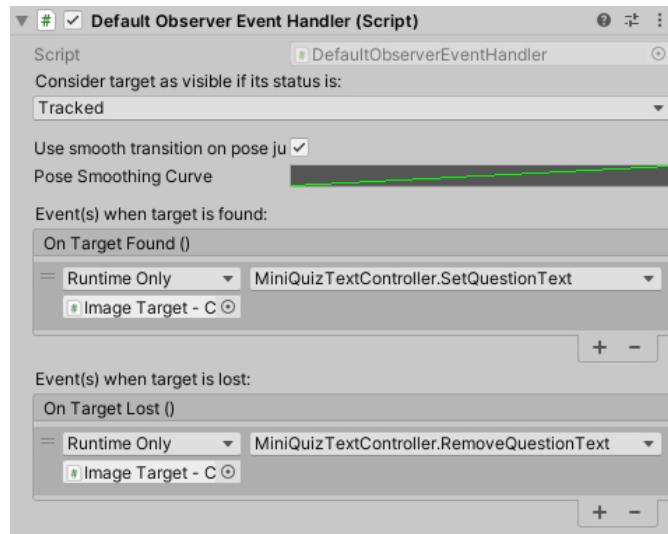
U svrhu zadovoljavanja funkcionalnih zahtjeva aplikacije, koji u trenutku implementacije novih modela još nisu bili ispunjeni, sastavljen je jednostavan sustav za označavanje pojedinih dijelova modela, čime je omogućeno odgovaranje na tematski postavljena pitanja unutar AR scene. Postupak razvoja ovakvog sustava uključivao je modifikaciju postojećeg objekta *User Interface* zaduženog za korisničke interakcije unutar virtualne scene te za prikaz tekstualnog sadržaja. Dodan je novi objekt *AR Mini Quiz Container* koji omogućuje prikazivanje pitanja za prikazani model, a proširena hijerarhija objekata korisničkog sučelja prikazana je na Slici 3.14.



Slika 3.14: Proširena hijerarhija objekata korisničkog sučelja uključujući *AR Mini Quiz Container*.

Na svaki od objekata *Image Target*, dodana je skripta *Mini Quiz Text Controller* koja omogućuje prikaz teksta pitanja te osvježavanje odabira objekata pozivom funkcije *ResetSelectedState()* (Programski isječak 3.20). Pomoću komponente *Default Observer Event Handler* postavljene su funkcije *OnTargetFound()* i *OnTargetLost()* koje uključuju odnosno isključuju tekst pitanja te osvježavaju stanje odabira ovisno o tome je li slikovna oznaka modela u fokusu. Postavke ovih komponenti prikazane su na Slikama 3.15 i 3.16.

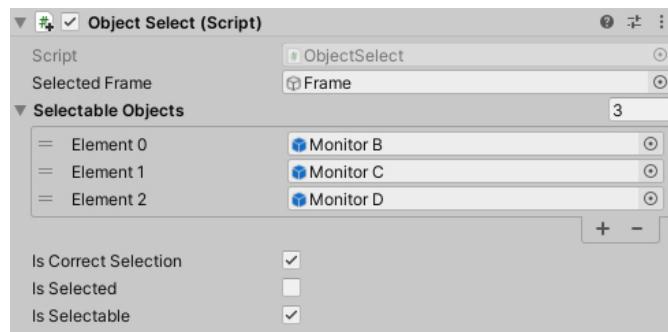
Svaki od dijelova modela koji je moguće označiti trebao je sadržavati komponentu *Mesh Collider* te skriptu *Object Select* kako bi korisnička interakcija bila moguća. Dodatno, svakom je takvom dijelu modela pridružen odgovarajući okvir koji se pojavljuje prilikom odabira, crvene ili zelene boje ovisno o točnosti odgovora. Slika 3.18 prikazuje okvir koji se pojavljuje oko dijela modela ako je on odabran i označen kao točan odgovor, a na Slici 3.17 je prikazan primjer postavki skripte *Object Select*.



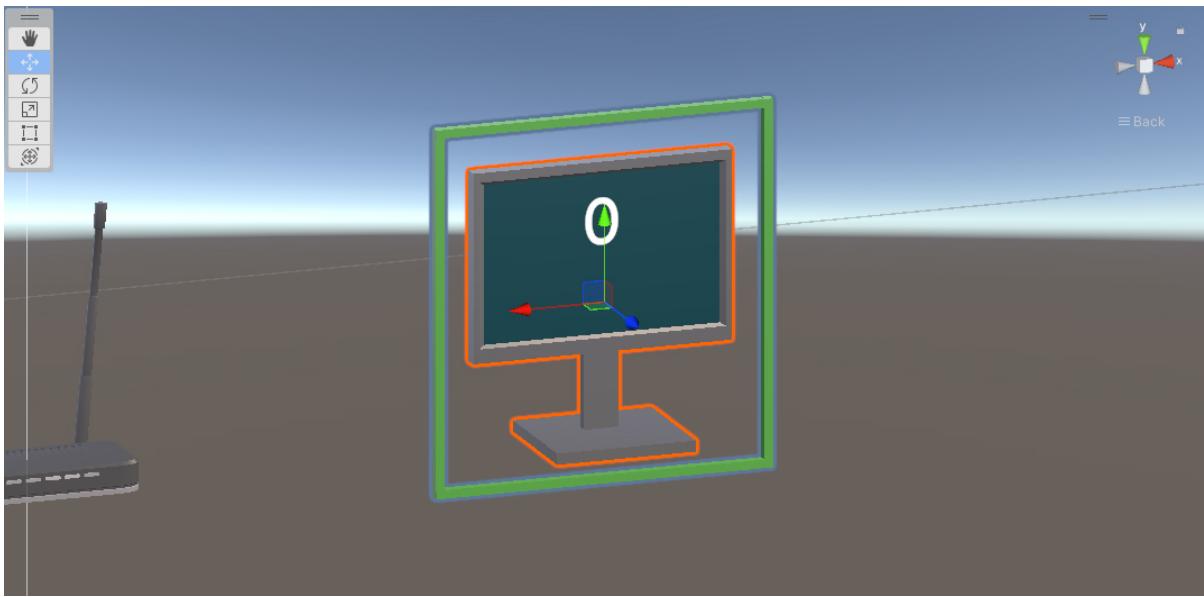
Slika 3.15: Postavke komponente *Default Observer Handler* zaslužne za prikaz teksta pitanja i osvježavanje stanja modela za odabir ovisno o fokusu slike označene.



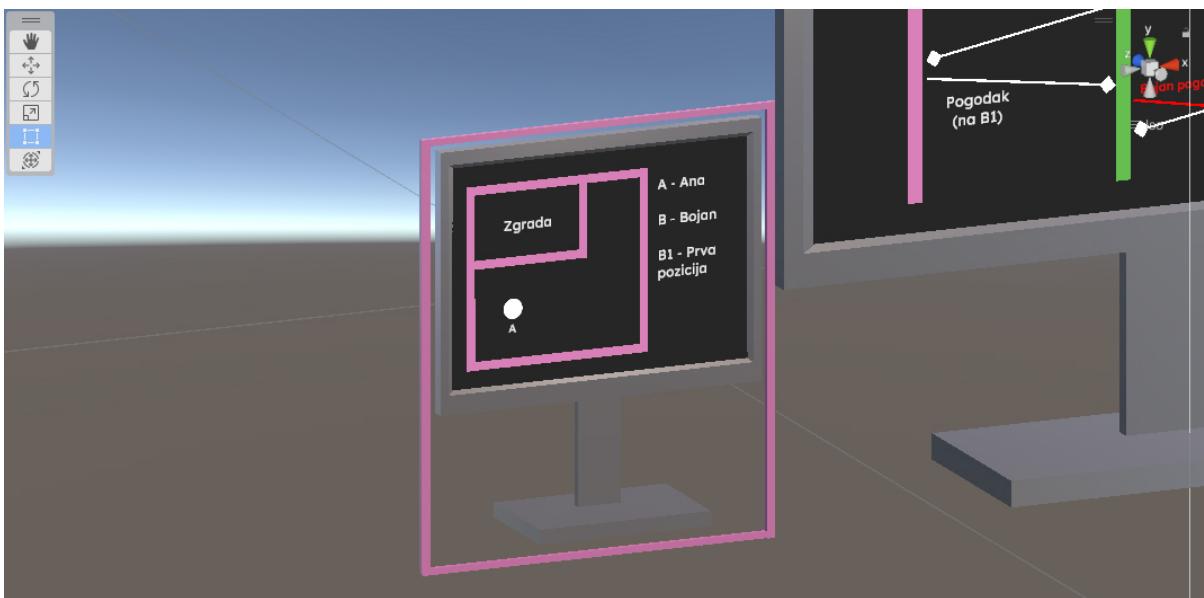
Slika 3.16: Postavke skripte *Mini Quiz Text Controller* zaslužne za prikaz teksta pitanja i osvježavanje stanja odabira modela kod prikaza slabe konzistentnosti.



Slika 3.17: Postavke skripte *Object Select* za dio modela koji je označen kao točan odgovor kod prikaza slabe konzistentnosti.



Slika 3.18: Okvir oko dijela modela koji je označen kao točan odgovor kod prikaza slabe konzistentnosti.



Slika 3.19: Okvir oko dijela modela koji je označen kao krivi odgovor kod prikaza slučaja "smrti iza zida".

Programski isječak 3.20 prikazuje funkcije *OnMouseDown()* i *ResetSelectedState()* iz skripte *Object Select*. Prilikom pritiska na objekt na kojem se skripta nalazi, provjerava se mogućnost odabira tog objekta te je li on već prethodno odabran. Sukladno tome, poduzimaju se akcije uključivanja ili isključivanja pripadajućeg okvira objekta, a *ResetSelectedState()* vraća postavke skripte na početne.

```

1     private void OnMouseDown() {
2         if (IsSelectable) {
3             _previouslySelected = !_previouslySelected;
4             switch (_previouslySelected) {
5                 case true:
6                     IsSelected = true;
7                     foreach (GameObject obj in SelectableObjects)
8                     {
9                         obj.gameObject.GetComponent<ObjectSelect>().IsSelected =
10                            false;
11                         if (IsCorrectSelection) {
12                             obj.gameObject.GetComponent<ObjectSelect>().
13                             IsSelectable = false;
14                         }
15                         break;
16                     case false:
17                         if (IsSelected && !IsCorrectSelection) {
18                             IsSelected = false;
19                         }
20                         break;
21                     default:
22                         break;
23                     }
24                 }
25             }
26         public void ResetSelectedState()
27         {
28             IsSelected = false;
29             IsSelectable = true;
30             _previouslySelected = false;
31         }
32     }

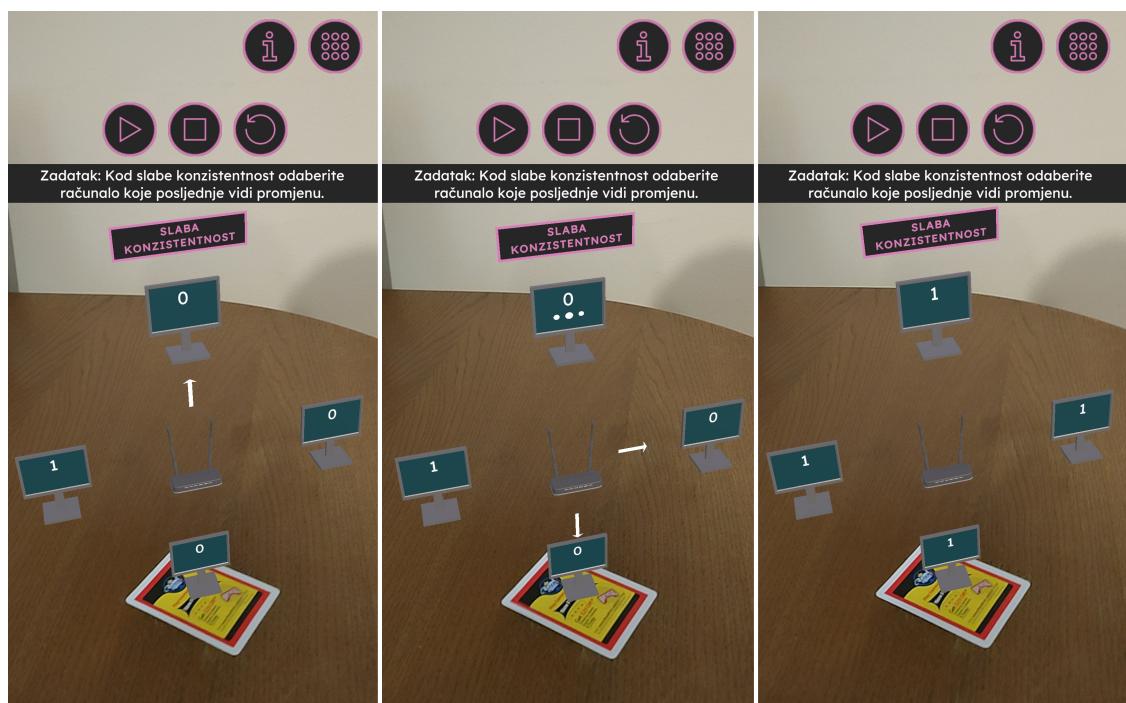
```

Programski isječak 3.20: Metode skripte *Object Select* za upravljanje stanjem odabira dijelova modela.

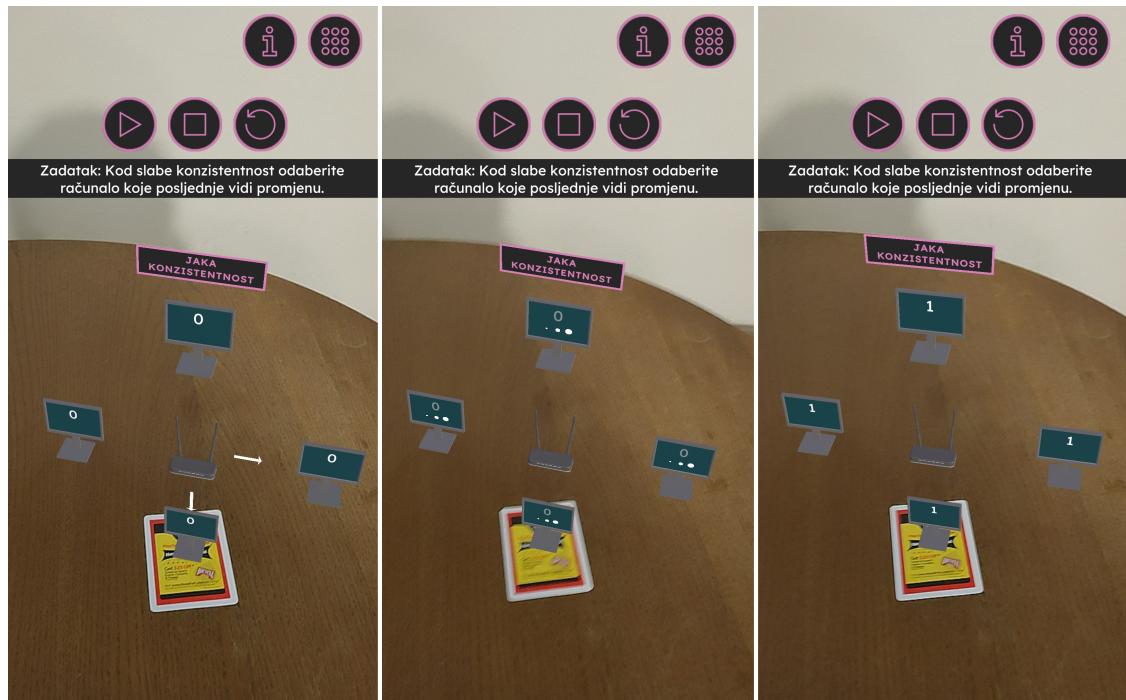
4. Rezultati i rasprava

U ovom su poglavlju prikazani konkretni rezultati implementacije. Prikazani su slikovni prikazi korištenja aplikacije skeniranjem oznaka u stvarnom prostoru te odgovaranje na postavljena pitanja označavanjem dijelova prikazanih modela AR scene. Također, predložena su poboljšanja u smislu modifikacija samih modela ili njihovih funkcionalnosti.

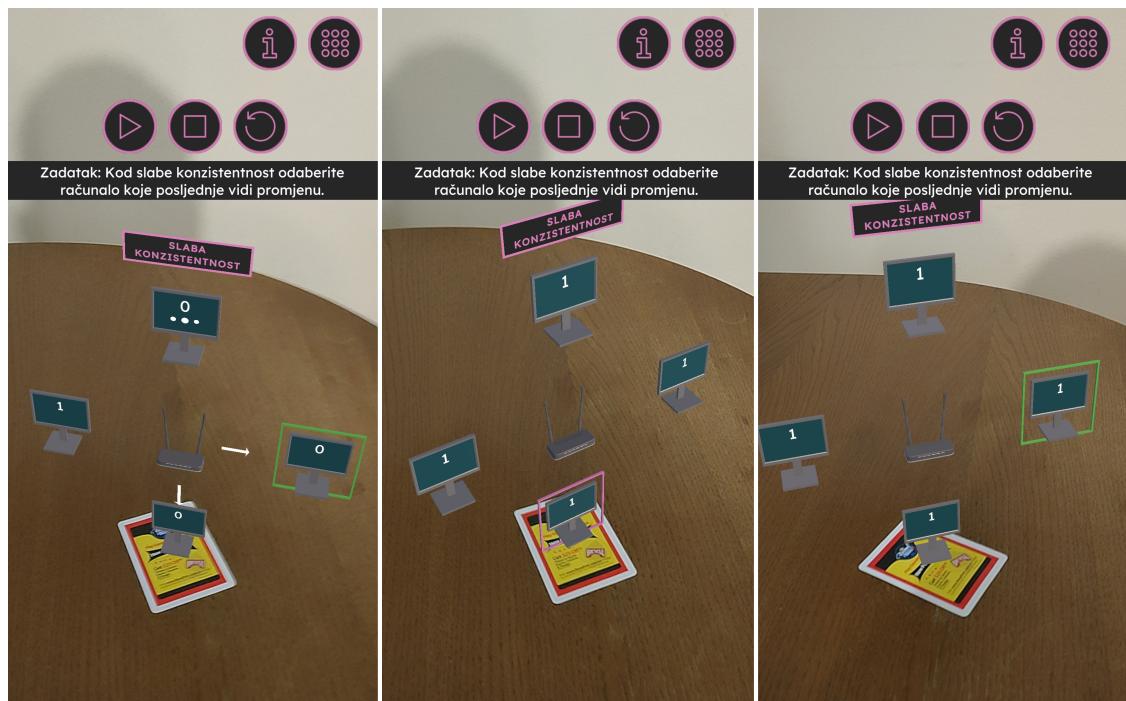
Skeniranjem slikovne oznake za prikaz modela konzistentnosti dobiveni su rezultati prikazani na Slikama 4.1 za slabu konzistentnost te 4.2 za jaku konzistentnost. Slike prikazuju specifične trenutke unutar animacija modela, a na Slikama 4.3 prikazan je postupak odabira dijela modela kao odgovor na postavljeno pitanje kod prikaza slabe konzistentnosti.



Slike 4.1: Rezultati skeniranja oznake za model slabe konzistentnosti.



Slike 4.2: Rezultati skeniranja oznake za model jake konzistentnosti.



Slike 4.3: Rezultati odabira dijela modela slabe konzistentnosti.

Skeniranjem slikovne označke za prikaz modela kašnjenja, odnosno slučaja "smrti iza zida", dobiveni su rezultati prikazani na Slikama 4.4.



Slike 4.4: Rezultati skeniranja oznake za model slučaja "smrti iza zida".

S obzirom na to da je sustav za označavanje dijelova modela implementiran isključivo pokazno i za testiranje, predlaže se daljnji razvoj i poboljšanje u smislu univerzalnosti i optimizacije. Cilj poboljšanja je lakša integracija za sve vrste modela, što uključuje zamjenu okvira oko dijela modela s nekom vrstom univerzalne naznake koju nije potrebno modificirati za svaki dio zasebno. Također, kao i za svaki interaktivni dio korisničkog sučelja, predlaže se dodavanje zvuka koji signalizira točan ili netočan odgovor, a moguće je dodavanje i sustava čestica koji dodatno označava uspjeh korisnika. Za svaki od implementiranih modела, moguće je optimizirati brzinu skaliranja, rotacije ili animacije te veličinu modela po želji.

5. Zaključak

Korištenje proširene stvarnosti u okviru edukativne aplikacije predstavlja značajan potencijal za povećanje motivacije korisnika za učenjem. Izravna interakcija s trodimenzionalnim modelima, unutar takvih aplikacija korisnicima omogućuje lakše utečmeljenje znanja kroz sudjelovanje u kvizovima. Neosporna je i praktičnost primjene sličnih aplikacija zbog njihove visoke dostupnosti i raspoloživosti na širem spektru mobilnih uređaja.

Unutar ovog je rada opisana implementacija dodatnih modela za postojeću AR aplikaciju koji kroz animacije prikazuju neka od svojstava umreženih videoigara te načine na koje ta svojstva utječu na korisničke sjednice igranja. Time se korisniku aplikacije, putem interakcije s virtualnom okolinom, pružaju informacije o važnosti stanja mreže za poboljšanje korisničkog iskustva kod umreženih videoigara.

Osim implementiranih modela, ostavlja se prostora i za prikaz drugačijih specifičnosti iz područja informacijske i komunikacijske tehnologije ili konkretnije raspodijeljenih sustava. Svođenjem komplikiranijih tehnoloških pojava na višu razinu apstrakcije, promatraču se pruža veća mogućnost razumijevanja koncepata iz raznolikih domena.

6. Literatura

- [1] L. V. Fernandes, C. D. Castanho, and R. P. Jacobi, “A survey on game analytics in massive multiplayer online games,” in *2018 17th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, pp. 21–2109, 2018.
- [2] S. C. McLoone, D. Wynne, A. B. McCoy, and T. E. Ward, “A consistency regulation algorithm for client-server-based multiplayer computer games,” in *IET Irish Signals and Systems Conference (ISSC 2012)*, pp. 1–6, 2012.
- [3] H. Rahimi, S. Ratti, A. A. N. Shirehjini, and S. Shirmohammadi, “Unsynchro-nized multiplayer networked games: Feasibility with time rewind,” in *2010 9th Annual Workshop on Network and Systems Support for Games*, pp. 1–2, 2010.
- [4] “Guide to network latency – how to check, measure, and reduce network latency.” <https://www.dnsstuff.com/network-latency>. 2019. Pristupljeno: 2023-04-20.
- [5] A. Sundji, “Razvoj edukacijske pokazne aplikacije za proširenu stvarnost,” Master’s thesis, University of Zagreb. Faculty of Electrical Engineering and Computing, July 2022.
- [6] M. Roccati, S. Ferretti, and C. E. Palazzi, “The brave new world of multiplayer online games: Synchronization issues with smart solutions,” in *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pp. 587–592, 2008.
- [7] N. Naik, “Comprehending concurrency and consistency in distributed systems,” in *2021 IEEE International Symposium on Systems Engineering (ISSE)*, pp. 1–6, 2021.
- [8] R. A. Golding, “A weak-consistency architecture for distributed information ser-vices,” *Computing Systems*, vol. 5, no. 4, pp. 379–405, 1992.

- [9] H. D., “System design interview concepts – eventual consistency,” 2018. Pristupljeno: 2023-04-21.
- [10] S. Burckhardt, “Consistency in distributed systems,” *Software Engineering: International Summer Schools, LASER 2013-2014, Elba, Italy, Revised Tutorial Lectures 10*, pp. 84–120, 2015.
- [11] C. N., “The no. 1 reason gamers will quit playing an online multiplayer game,” 2019. Pristupljeno: 2023-04-22.

7. Sažetak

U ovom je radu predstavljena primjena tehnologije proširene stvarnosti za edukaciju o svojstvima umreženih videoigara. Nakon kratkog pregleda tematike mrežnih svojstava kod umreženih videoigara, posebna je pažnja posvećena konzistentnosti i kašnjenju u mreži. Opisan je postupak implementacije dvaju 3D modela koji prikazuju spomenuta svojstva unutar postojeće AR aplikacije za edukaciju učenika u školama. Animacija mrežne konzistentnosti prikazuje proces promjene stanja jednog računala, prelazak na mrežu te širenje promjene do ostalih korisnika mreže. Drugi model prikazuje dijagram slučaja smrti u videoigri s razlikama u kašnjenju dvaju računala povezanih na isti poslužitelj. U radu se iznose detalji implementacije, a na posljetku su prikazani i rezultati uz raspravu. Ovaj rad ukazuje na mogućnosti primjene AR tehnologije u edukaciji o svojstvima umreženih videoigara te općenito podržava novi pristup učenju o kompleksnim tehnološkim konceptima.