Izboljšanje učinkovitosti semaforiziranih križišč s simulacijami in strojnim učenjem

Šemso Hrnjičić, Uroš Rajkovič
Univerza v Mariboru, Fakulteta za organizacijske vede, Kidričeva cesta 55a, 4000 Kranj
semso.hrnjicic@gmail.com, uros.rajkovic@um.si

Izvleček

Razvoj simulacijskega okolja za proučevanje prometnih sistemov omogoča globlje razumevanje in boljše reševanje izzivov v prometu. V raziskavi smo se osredotočili na optimizacijo semaforiziranega križišča s ciljem izboljšanja njegove učinkovitosti in zmogljivosti. S pomočjo najsodobnejših orodij, kot sta Unity in Blender, smo ustvarili dinamičen model, ki lahko simulira različne prometne scenarije in se odziva na realno časovne spremembe v prometnem okolju. Zbiranje in analiza podatkov simulacije nam je omogočila, da natančno prilagodimo časovne cikle semaforjev in implementiramo inteligentne prometne nadzorne sisteme. Z integracijo naprednih tehnologij strojnega učenja smo razvili nevronske mreže, ki optimizirajo prometno signalizacijo in dramatično zmanjšajo čakalne čase. Rezultati naše raziskave kažejo občutne izboljšave v pretočnosti in varnosti prometa, kar dokazuje, da je pristop z uporabo simulacij in implementacijo simuliranih optimizacij ključen za prihodnje izboljšave v urbanem prometnem načrtovanju.

Ključne besede: simulacija križišča, spodbujevalno učenje, nevronske mreže, optimizacija

Improving the efficiency of signalized intersections with simulations and machine learning

Abstract

The development of a simulation environment for the study of transport systems enables a deeper understanding and a better solution to traffic challenges. In the research, we focused on the optimization of a traffic light intersection with the aim of improving its efficiency and performance. Using state-of-the-art tools such as Unity and Blender, we have created a dynamic model that can simulate various traffic scenarios and respond to real-time changes in the traffic environment. The collection and analysis of simulation data allowed us to precisely adjust traffic light cycles and implement intelligent traffic control systems. By integrating advanced machine learning technologies, we have developed neural networks that optimize traffic signals and dramatically reduce waiting times. The results of our research show significant improvements in traffic flow and safety, proving that the approach using simulations and the implementation of simulated optimizations is crucial for future improvements in urban traffic planning.

Keywords: Intersection simulation, reinforcement learning, neural network, optimisation

1 UVOD

Raziskovanje optimizacije pretočnosti prometa na prometnih križiščih z uporabo simulacijskih modelov je pomembno za razumevanje in izboljšanje pretočnosti prometa ter zmanjšanje zastojev. Semaforizirana križišča predstavljajo kompleksne sisteme, kjer je usklajevanje pretoka vozil ključnega pomena za preprečevanje prometnih zastojev in izboljšanje prometne varnosti. Simulacijski modeli lahko učinkovito ponazarjajo realne prometne razmere in pomagajo pri analizi različnih scenarijev.

Razvoj simulacijskega okolja za izboljšavo sema-

foriziranega križišča je aktualna tema zaradi naraščajoče urbanizacije in prometnih obremenitev, ki zahtevajo učinkovite rešitve za zmanjšanje zastojev, čakalnih časov in onesnaževanja. Simulacijska okolja omogočajo varno in stroškovno učinkovito testiranje različnih optimizacijskih pristopov, brez neposrednega poseganja v realni promet. Tako omogočajo izboljšanje pretočnosti prometa in podpirajo načrtovanje trajnostnih prometnih sistemov, hkrati pa zagotavljajo dragocene vpoglede za izboljšanje obstoječe prometne infrastrukture v mestih.

Optimizacija semaforiziranih križišč ni nič novega. Promet optimiziramo na različne načine in tudi drugi raziskovalci eksperimentirajo z optimizacijo s pomočjo strojnega učenja. Iz ene od relevantnih raziskav izhaja sledeče:

Eksperimentalni rezultati kažejo, da lahko sistemi s spodbujevalnim učenjem prekašajo številne prilagodljive sisteme. Eden od sistemov, TC-3, uporablja globalno komunikacijo med semaforji in lahko preseže delovanje drugih algoritmov, kadar promet ni zelo gost. Če je prometno omrežje nasičeno, ko povečujemo prometno obremenitev, sistemi s spodbujevalnim učenjem očitno presegajo fiksne krmilnike in tudi veliko profitirajo od sočasnega učenja. [1]

V primeru nepričakovanih zastojev (na primer zaradi nesreč) obveščeni potniki skrajšajo svoj potovalni čas z zamenjavo poti, vendar posledično alternativne poti postanejo polnejše, zaradi česar se lahko podaljšujejo potovalni časi za neobveščene voznike. [2]

Do danes so bile raziskane tako tradicionalne kot sodobne metode optimizacije semaforiziranih križišč, pri čemer raziskave z uporabo umetne inteligence ponujajo potencial za prilagodljivejše in učinkovitejše delovanje semaforiziranih križišč. Sodobne raziskave kažejo tudi na to, da boljši senzorji in vedno večja količina podatkov ne pomenijo vedno boljšo stopnjo izboljšave pretoka semaforiziranega križišča.

Rezultati kažejo, da podatki, zbrani iz tradicionalnih in vseprisotnih senzorjev, kot so detektorji zanke, zadostujejo za spodbujevalno učenje. Po modelu, oblikovanem v tej raziskavi, predstavitve stanja visoke ločljivosti, ki zahtevajo prefinjene senzorje, nudijo izboljšave le z zmanjšanjem zakasnitve za približno 10 %, brez razlike v čakalni vrsti ali pretoku. [3]

V okviru raziskave smo razvili simulacijski model semaforiziranega križišča z uporabo platforme Unity in simulirali pretok križišča z realnimi podatki. Nato smo analizirali simuliran pretok in uporabili tri različne metode optimizacije za izboljšanje učinkovitosti in natančnosti modela. V nadaljevanju bomo predstavili:

- vizualizacijo semaforiziranega križišča,
- podatke simulacije,
- uporabljene tehnike za optimizacijo semaforiziranega križišča in
- primerjavo rezultatov različnih metod optimizacije, da bi ocenil njihovo uspešnost.

Najprej bomo podrobneje predstavili vizualizacijo semaforiziranega križišča, nato bomo predstavili, kako delujejo vgrajeni senzorji, kako smo obdelali in uporabili pridobljene podatke, na koncu pa kako smo uporabili tehnike spodbujevalnega učenja za izboljšanje sistema. Prav tako bomo analizirali in primerjali rezultate različnih pristopov k optimizaciji križišča, kar nam bo dalo jasno sliko uspešnosti posameznih metod.

2 METODOLOGIJA

V naši raziskavi smo kot osnovni pristop uporabili načrtovanje in razvoj (Design Science Research, Hevner et al., 2004). V okviru tega pristopa razvijamo artefakt – simulacijski model semaforiziranega križišča – skozi tri zaporedne cikle:

- Cikel strogosti

V tej fazi smo izvedli temeljit pregled obstoječe literature in metod, ki se ukvarjajo s problematiko semaforiziranih križišč. Za izvedbo temeljitega pregleda literature smo opravili iskalni postopek v akademskih bazah ter pregledali druge vire (konferenčni zborniki, tehnična poročila, magistrske in doktorske naloge). Iskalne pojme smo oblikovali okoli ključnih tem (npr. *traffic signal optimization, traffic simulation, intersection modelling, adaptive traffic control, traffic light intersection, Unity simulation*), pregledali naslove in izvlečke, nato pa celotna besedila študij, ki so ustrezala našim kriterijem pregledali (študije, ki obravnavajo optimizacijo semaforiziranih križišč, modeliranje in simulacijo prometa, empirične meritve ali eksperimentalno vrednotenje). Ta postopek je zagotavljal pokritost teorije, metodologij in praktičnih primerov, relevantnih za naš raziskovalni cilj. Na podlagi te analize smo prepoznali pomembnost optimizacije semaforiziranih križišč in se odločili, da kot raziskovalni predmet izberemo semaforizirano križišče v Kranju.

- Cikel relevantnosti

Namen te faze je preveriti, ali ima optimizacija izbranega križišča praktično utemeljitev v realnem okolju. Pred izvedbo optimizacijskega postopka smo si zastavili ključno vprašanje: »Ali je izboljševanje delovanja semaforiziranega križišča smiselno?«. Z namenom odgovora na to vprašanje smo razvili simulacijski model, s katerim smo ocenili vpliv morebitnih sprememb na prometni pretok in varnost na križišču ter v njegovi okolici.

- Cikel razvoja

V tej fazi smo razvili in implementirali simulacijski model semaforiziranega križišča. Model smo izdelali v 3D okolju Unity, pri čemer smo uporabili empirično zbrane podatke – meritve pretoka na vseh vhodih križišča, izvedene v obdobju ene ure. Poleg funkcionalne simulacije smo zagotovili tudi vizualno predstavitev, da bi si lažje predstavljali stanje na križišču in njegovem neposrednem okolju. Na podlagi razvitega modela smo izvedli vrsto eksperimentov, kjer smo optimizirali križišče na tri različne načine. Primerjava eksperimentalnih rezultatov nam je omogočila oceno učinkovitosti posameznih optimizacijskih pristopov in določitev morebitnega profita izboljšav.

Z navedenim celovitim pristopom smo dosegli tako teoretično kot praktično podlago za nadaljnje raziskave in morebitno implementacijo optimizacij na izbranem semaforiziranem križišču.

3 REZULTATI

Za robustno povezavo simulacijskega modela z realnim stanjem smo izvedli empirično meritev pretoka vozil: enourno štetje vozil na vseh vhodih semaforiziranega križišča pri trgovskem centru Tuš Supermarket Planet Kranj (presečišče cest Cesta Boštjana Hladnika in Cesta Rudija Šeliga) je bilo izvedeno v soboto, 23. oktobra 2021. Pridobljene podatke smo uporabili za kalibracijo vhodnih pretokov in drugih parametrov modela ter kot osnovo za izvedbo eksperimentov in primerjavo rezultatov optimizacijskih scenarijev. Ta empirična podlaga zagotavlja, da so eksperimentalni scenariji zasnovani glede na dejanske prometne pogoje opazovanega križišča.

V tem poglavju predstavljamo ključne rezultate raziskave, ki zajemajo razvoj in uporabo simulacijskega modela izbranega semaforiziranega križišča ter vrednotenje učinkovitosti različnih optimizacijskih pristopov. Najprej podamo razvoj vizualnega dela raziskave, ki omogoča boljše razumevanje prometnih razmer in postavitve križišča. Nato opišemo postopek razvoja simulacijskega modela, vključno z uporabo empiričnih podatkov in tehničnimi vidiki implementacije. V zadnjem delu predstavimo rezultate optimizacijskih scenarijev, primerjamo njihove učinke ter ocenimo potencialne izboljšave prometnega toka in varnosti.

3.1 Vizualizacija

Vizualizacija je izjemno pomembna komponenta raziskave, ki olajša razumevanje problemov in omogoča boljše predstavljanje potrebnih sprememb in modifikacij. Za dosego slednjega cilja smo se odločili uporabiti programsko orodje Blender, ki nam je omogočilo natančno izdelavo tridimenzionalnega modela izbranega križišča. Po izdelavi smo model brez težav uvozili v program Unity, ki smo ga uporabili za nadaljnji razvoj simulacije.

Za bolj realistično upodobitev simulacijskega okolja smo v simulacijo vključili tudi satelitske posnetke območja, ki služijo kot ozadje za križišče. Dodali smo modele avtomobilov, ki predstavljajo raznolik promet, in modele semaforjev, ki regulirajo prometne tokove. Da bi dosegli še večjo verodostojnost, smo semaforjem dodali funkcionalne luči – zelene, rumene in rdeče, kar omogoča prikaz različnih faz prometne signalizacije.

Unity omogoča uvoz in sestavljanje sredstev, pisanje kode za interakcijo s predmeti, ustvarjanje ali uvoz animacij za uporabo z naprednim animacijskim sistemom in še veliko več. [4]

Vse te elemente lahko opazimo na sliki 1, ki prikazuje končni izgled vizualizacije križišča. Ta vizualizacija služi ne samo kot demonstracija funkcionalnosti modela, ampak tudi kot sredstvo za preverjanje, kako različni elementi koordinirano delujejo.

3.2 Razvoj simulacijskega modela

Za namen izvedbe simulacije našega semaforiziranega križišča smo razvili dve ključni entiteti, vsako s svojo specifično logiko in funkcionalnostjo, ki sta nujni za realistično in učinkovito modeliranje prometnih tokov.

Skripta mora biti pripeta na objekt v sceni, da ga Unity lahko uporabi. Skripte so napisane v posebnem jeziku, ki ga Unity razume. Prek tega jezika se lahko pogovarjamo z motorjem in mu dajemo navodila. Jezik, ki se uporablja v Unity, se imenuje C#. Vsi

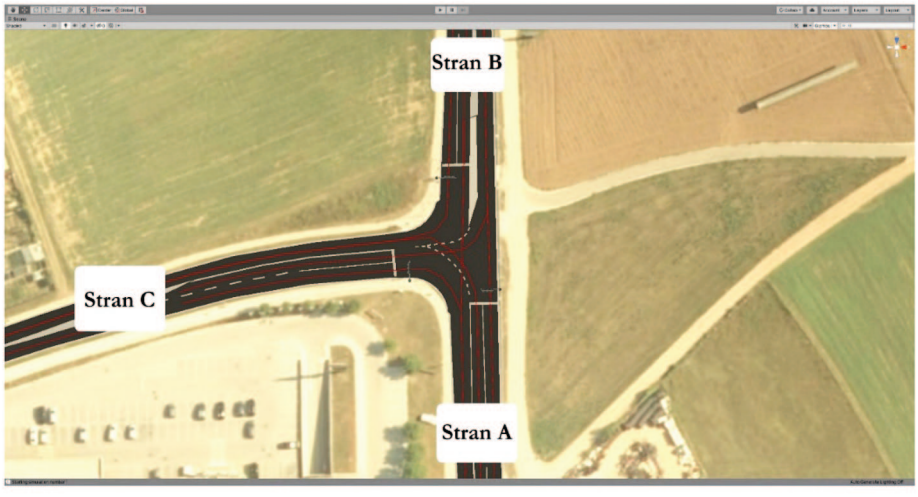
Slika 1: **Vizualizacija križišča**

jeziki, s katerimi Unity deluje, so objektno usmerjeni skriptni jeziki. Kot vsak jezik imajo skriptni jeziki sintakso ali dele govora, primarni deli pa se imenujejo spremenljivke, funkcije in razredi. [5]

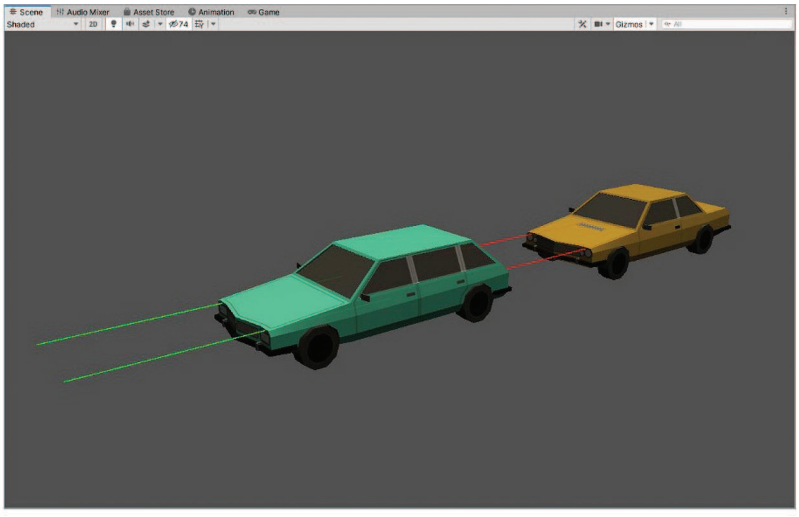
C# je preprost, sodoben, objektno usmerjen in tipno varen programski jezik. Ima svoje korenine v družini jezikov C in je znan predvsem programerjem C, C++ in Java. C# je ECMA International standardiziral kot standard ECMA-334, ISO/IEC pa kot standard ISO/IEC 23270. [6]

Prva entiteta je semaforizirano križišče, ki deluje kot nadzorna entiteta, ki ves čas simulacije skrbno spremlja čas in prometne razmere. Na podlagi predhodno določenih spremenljivk, kot so gostota prometa in posebne situacije na cesti, križišče dinamično določa, katera smer bo imela prednost. Sistem semaforjev je programiran tako, da maksimira pretočnost in minimizira čakalne čase, s čimer pripomore k večji varnosti in učinkovitosti na cesti.

Na sliki 2 je prikazan zračni pogled na izbrano križišče, ki nam daje jasno predstavo o njegovi strukturi in razporeditvi. Križišče je razdeljeno na tri različne sektorje: Stran A, Stran B in Stran C. Strani A in B predstavljata glavne, prednostne ceste, medtem ko je Stran C kategorizirana kot stranska cesta.



Slika 2: **Strani križišča**



Slika 3: **Vizualizacija senzorjev**

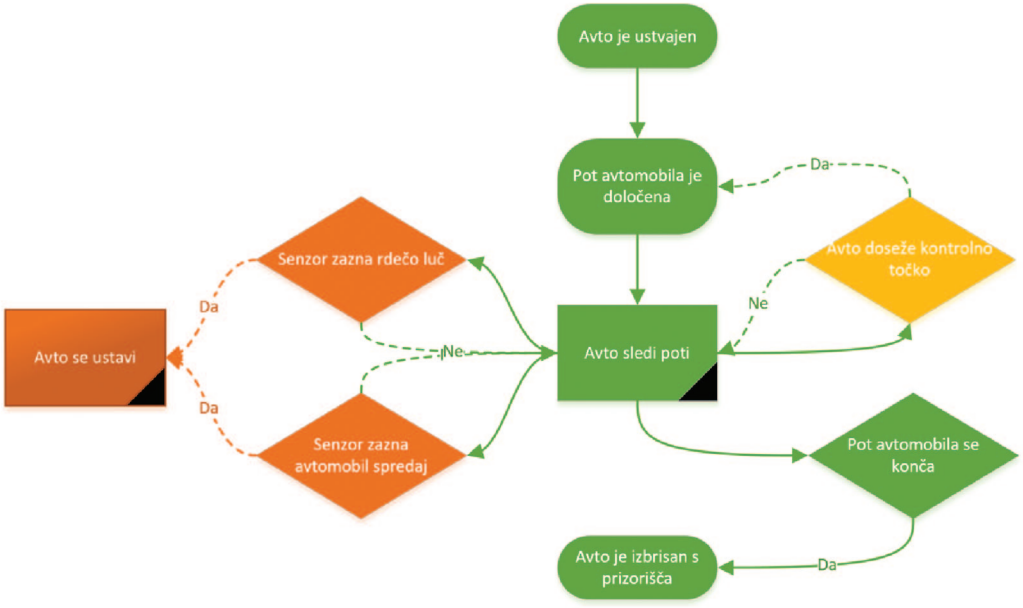
Druga entiteta je posamezen avtomobil. Vsak avtomobil v simulaciji predstavlja ločeno entiteto, ki se zaveda svoje lokacije in cilja. Avtomobili se vozijo proti križišču, kjer ob prihodu na določeno kontrolno točko sprejmejo odločitev o smeri nadaljevanja - levo, desno ali naravnost. Logika, ki usmerja odločitve avtomobilov, temelji na algoritmih, ki so bili razviti na podlagi zbranih podatkov o prometu. Ta logika omogoča avtomobilom, da reagirajo na spremenljive prometne razmere in semaforizacijo, kar zagotavlja pretočnost prometa in zmanjšuje tveganje za zastoje.

Da bi zagotovili, da vozila v simulaciji spoštujejo cestnoprometne predpise in se odzivajo na prometne razmere na realističen način, smo v modele vozil implementirali senzorje. Ti senzorji omogočajo zaznavanje drugih vozil, kontrolnih točk in prednostnih točk na cestišču. S tem smo vozilom omogočili, da se inteligentno odzivajo na spremembe v okolju, kot so spremembe v prometni luči ali nepredvidene ovire na cesti, kar povečuje realnost naše simulacije. Ta pristop ne samo izboljša natančnost simulacije, temveč tudi poveča njeno pedagoško vrednost pri demonstraciji, kako dinamični sistemi, kot so semaforizirana križišča, vplivajo na urbani promet.

Na sliki 3 je prikazan primer dveh avtomobilov, ki ilustrira delovanje vgrajenih senzorjev. Avtomobil ne zazna nobenih ovir pred seboj, kar je označeno z žarki senzorja zelene barve, ki ponazarjajo varno pot. Senzorji avtomobila na desni strani pa nasprotno zaznajo prisotnost drugega avtomobila neposredno pred seboj. Žarki senzorja se zato obarvajo rdeče, kar kaže na nevarnost trka ali potrebo po zmanjšanju hitrosti. Informacija o zaznani oviri se samodejno posreduje nadzornemu sistemu vozila, ki nato ustrezno prilagodi vedenje avtomobila, da ta izogne morebitni nevarnosti.

Diagram logike delovanja avtomobilov, ki je prikazan na sliki 4, podrobno opisuje procese, skozi katere avtomobil potuje od začetka do konca svoje poti v simuliranem okolju. Vsak avtomobil začne s specifičnim ciljem, do katerega je usmerjen z nizom vnaprej določenih kontrolnih točk. Ko avtomobil doseže kontrolno točko, prejme informacije o naslednjem cilju. Med potovanjem so avtomobili opremljeni s senzorji, ki nenehno ocenjujejo okoliško situacijo in omogočajo avtomobilu, da ustavi v treh specifičnih scenarijih: če zazna rdečo luč, če je neposredno pred njim drugo vozilo ali če mora zaviti in čaka na prednost nasproti vozečih vozil. Ko avtomobil doseže končni cilj, je iz scenarija odstranjen, da se ohrani optimalna zmogljivost simulacije in prepreči nepotrebna zasičenost prizorišča.

Za natančne analize prometnega toka smo merili gostoto prometa na tem križišču, ki je trajalo eno uro. V tem času smo zabeležili, da je skozi Stran A prešlo 690 vozil, kar jo uvršča kot najbolj obremenjeno stran križišča. Stran B je imela prav tako visok pretok z 646 vozili na uro, medtem ko je Stran C, kot manj prometna stranska cesta, zabeležila precej nižji prometni tok, s samo 120 vozili na uro. Te podatke prikažemo v tabeli 1, kjer je prikazano število avtomobilov, ki so prečkali križišče z vsake smeri v času ene ure. Takšna kvantitativna analiza prometa nam omogoča, da bolje razumemo dinamiko pretoka in vpliv razporeditve prometa na delovanje križišča.



Slika 4: **Logika delovanja avtomobila v simulaciji**

Tabela 1: **Podatki pretoka križišča**

	Število avtomobilov/h	Levo/h	Naprej/h	Desno/h	Levo v %	Naprej v %	Desno v %
A	690	108	582	0	16	84	0
B	646	0	564	82	0	87	13
C	120	84	0	36	70	0	30

Po zaključku razvoja našega simulacijskega modela smo se lotili uporabe zbranih podatkov za simulacijo prometnega toka na semaforiziranem križišču. Obdobje simulacije je trajalo eno uro. Da bi zagotovili zanesljivost naših rezultatov, smo vsako posebno stanje na križišču simulirali desetkrat, nato pa smo izvedli analizo povprečnih vrednosti, ki so bile zabeležene. Rezultate teh simulacij smo strnili v tabeli 2, kjer so povprečni rezultati prikazani v formatu ur, minut in sekund (hh:mm:ss). Posebno pozornost smo namenili analizi skupnih čakalnih časov avtomobilov za vsako smer posebej, da bi ugotovili, katera smer križišča povzroča najdaljše zastoje. Ugotovili smo, da avtomobili, ki prihajajo iz smeri A, v povprečju najdlje čakajo na semaforju, in sicer približno uro in 37 sekund.

Tabela 2: **Povprečni čakalni čas avtomobilov**

hh:mm:ss			
Čakalni čas A	Čakalni čas B	Čakalni čas C	Skupni čakalni čas
01:00:37	00:57:04	00:51:36	02:49:18

Slednji podatek je ključnega pomena za nadaljnje ukrepanje pri optimizaciji križišča, saj izpostavlja specifične izzive, s katerimi se srečujemo na najbolj obremenjeni smeri križišča. Razumevanje vzorcev čakanja nam omogoča ciljni pristop k izboljšavam načrtovanja semaforizacije in tako izboljšanje pretoka prometa na kritičnih točkah.

3.3 Optimizacija

V fazi optimizacije smo se osredotočili na tri glavne pristope za izboljšanje delovanja semaforiziranega križišča. Prvi pristop je vključeval prilagoditev časovnih intervalov semaforja. S pomočjo našega simulacijskega modela smo preizkusili različne kombinacije časov za prednostne in stranske ceste. Ta metoda nam je omogočila hitro in stroškovno učinkovito eksperimentiranje, kar je bistveno za iskanje optimalne nastavitve semaforjev, ki bi izboljšala pretočnost prometa.

Drugi pristop je bil implementacija senzorjev, ki omogočajo spremljanje trenutnih prometnih razmer v realnem času. S pomočjo teh senzorjev smo sema-

foriziranemu križišču omogočili, da dinamično prilagaja svoje delovanje glede na spremenljive prometne pogoje. To je pripomoglo k večji prilagodljivosti sistema in boljšemu odzivanju na nepričakovane spremembe v prometnih tokovih.

Tretji način optimizacije je vključeval uporabo strojnega učenja. Razvili smo model nevronske mreže s pomočjo odprtokodnega projekta ML-Agents in odprtokodnega okolja TensorFlow, ki je omogočil, da se naše simulacijsko križišče uči iz zbranih podatkov in samodejno prilagaja svoje delovanje z namenom izboljšanja pretočnosti in skrajšanja čakalnih časov.

Vse tri optimizacije smo temeljito preizkusili in skrbno dokumentirali rezultate. Povprečne vrednosti iz različnih simulacijskih scenarijev smo uporabili za analizo in primerjavo učinkovitosti posameznih pristopov. Na ta način smo lahko določili, katera metoda najbolj učinkovito izboljša prometno situacijo na križišču.

3.3.1 Strojno učenje

Strojno učenje uporabljamo kot tretji način optimizacije delovanja semaforiziranega križišča. Odločili smo se uporabiti paket ML-Agents, ki omogoča, da naš simulacijski model postane inteligenten agent, zmožen samostojnega učenja in odločanja.

Zbirka orodij ML-Agents je obojestransko koristna tako za razvijalce iger kot za raziskovalce umetne inteligence, saj zagotavlja osrednjo platformo, na kateri je mogoče oceniti napredek umetne inteligence v bogatih okoljih Unity in nato omogočiti dostop širši skupnosti raziskovalcev in razvijalcev iger. [7]

K navedenemu pristopamo s tehniko spodbujevalnega učenja. Spodbujevalno učenje je močno orodje, ki izhaja iz principov živalske in vedenjske psihologije, in ima ključno vlogo na mnogih področjih strojnega učenja. Ta pristop se osredotoča na interakcijo med agentom in okoljem, kjer agent prejema bodisi nagrade bodisi kazni, odvisno od učinka njegovih dejanj na okolje. V začetnih fazah učenja agent eksperimentira z različnimi strategijami, njegovo obnašanje pa se sčasoma oblikuje na podlagi povratnih informacij, ki jih prejme. To pomeni, da agent prejme nagrade, če njegova dejanja privedejo do pozitivnih izidov. Nagrade ga spodbujajo k ponovitvi uspešnih dejanj. Nasprotno, negativne nagrade ali kazni ga odvračajo od morebitnih škodljivih ali neučinkovitih dejanj.

Slika 5: **Elementi spodbujevalnega učenja**

Navedeni proces omogoča agentu, da se uči in prilagaja, kar povečuje njegovo sposobnost sprejemanja optimalnih odločitev v danih situacijah. Spodbujevalno učenje se uporablja v številnih aplikacijah, od iger, kjer se agenti učijo igrati zapletene igre na visoki ravni, do industrijskih aplikacij, kjer sistemi samodejno optimizirajo delovanje naprav za zmanjševanje porabe energije ali izboljšanje proizvodnje.

Glavni cilj našega učenja je minimiziranje čakalnih časov za vozila, kar je bistveno za izboljšanje učinkovitosti prometnega toka in zmanjšanje zastojev. Uvedba spodbujevalnega učenja ni le izvedbena prednost, ampak tudi inovativni pristop k prometni regulaciji, ki presega tradicionalne metode upravljanja semaforjev. Spodbujevalno učenje omogoča agentu, da iz iteracij učenja razvije strategije, ki optimizirajo dodeljevanje zelenih faz v realnem času glede na trenutne pogoje prometa.

Spodbujevalno učenje je eno najbolj vznemirljivih in tudi eno najstarejših področij strojnega učenja danes. Obstaja že od petdesetih let dvajsetega stoletja in je proizvedlo veliko zanimivih aplikacij, zlasti na področju iger (na primer TD-Gammon, Backgammon-playing program) in na področju nadzora strojev, vendar se je o teh aplikacijah redko pisalo na naslovnicah. Leta 2013 pa se je zgodila revolucija, ko so raziskovalci britanskega podjetja DeepMind predstavili sistem, ki bi se lahko naučil igrati skoraj katero koli igro konzole Atari in je sčasoma postal celo boljši od ljudi. Pri svojem delovanju uporablja zgolj neobdelane slike kot vhod brez predhodnega poznavanja pravil igre. [8]

Na sliki 6 prikazujemo, kako naš semafor (Agent), opremljen z mrežo senzorjev, nenehno spremlja stanje na križiščih (Okolje) in kako se na podlagi zbranih

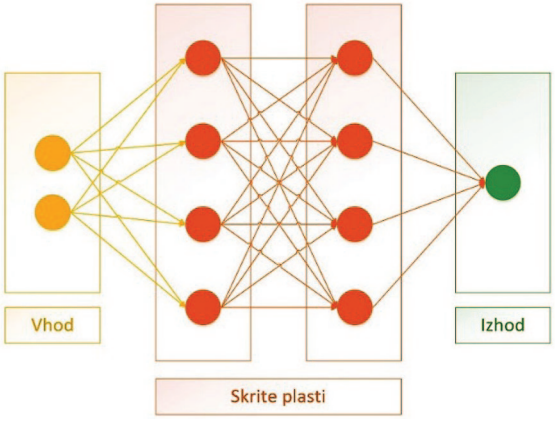
podatkov samostojno odloča, kateri smeri bo dodelil prednost. Povratne informacije, ki jih agent prejema od simulacijskega okolja, so oblikovane kot nagrade ali kazni. Te so neposredno povezane z izmerjenimi čakalnimi časi, kjer pozitivne nagrade stimulirajo tiste odločitve, ki skrajšajo čakalne dobe, medtem ko negativne kazni odvračajo od odločitev, ki vodijo v daljša čakanja. Ta sistem nagrajevanja in kaznovanja omogoča postopno izboljševanje strategij semaforizacije, s čimer povečujemo pretočnost in zmanjšujemo zastoje, kar ima za posledico izboljšano splošno učinkovitost križišča.

Slika 6: **Dinamika okolja in agenta**

Tako se skozi proces spodbujevalnega učenja naš model ne samo nauči odzivati na neposredne prometne izzive, temveč tudi predvideva in proaktivno upravlja prometne tokove, kar omogoča bolj tekoče in varnejše prometne pogoje na kritičnih urbanih točkah.

Na sliki 7 je prikazana končna konfiguracija nevronske mreže, ki bo uporabljena za dejansko delovanje v naši raziskavi. Ta mreža bo imela tri skrite plasti, pri čemer vsaka plast vsebuje 256 enot. Vizualna predstavitev modela jasno ločuje različne komponente: vhodne, skrite in izhodne plasti. Vhodna plast, označena z rumeno barvo, sprejema podatke iz okolja, ki so lahko različni prometni parametri, kot so gostota prometa, čas dneva, vremenske razmere in drugi relevantni dejavniki. Te vhodne podatke model uporabi za procesiranje v skritih plasteh nevronske mreže.

V modelu, kot je prikazan na sliki, so skrite plasti, ki so ključne za učenje in obdelavo vhodnih informacij, med vhodom in izhodom. Vsaka skrita plast vsebuje enote (nevrone), ki so prikazane kot rdeči krogi. Te enote delujejo kot procesorski elementi, ki težijo k preoblikovanju vhodnih podatkov v smiselne izhodne signale. V tem primeru imamo dve skriti plasti, kjer vsaka plast vključuje štiri enote. Ta konfiguracija omogoča mreži, da ujame in obdela kompleksne vzorce v podatkih, kar je bistveno za učinkovito odločanje o prometnih signalizacijah.



Slika 7: **Model nevronske mreže**

Z uporabo te konfiguracije nevronske mreže lahko pričakujemo izboljšanja v odzivanju križišč na spremenljive prometne pogoje, kar bo vodilo do

```yaml
default:
    trainer: ppo
    batch_size: 1024
    beta: 5.0e-3
    buffer_size: 10240
    epsilon: 0.2
    hidden_units: 128
    lambd: 0.95
    learning_rate: 3.0e-4
    learning_rate_schedule: linear
    max_steps: 5.0e5
    memory_size: 128
    normalize: false
    num_epoch: 3
    num_layers: 2
    time_horizon: 64
    sequence_length: 64
    summary_freq: 10000
    use_recurrent: false
    vis_encode_type: simple
    reward_signals:
        extrinsic:
            strength: 1.0
            gamma: 0.99

brain:
    batch_size: 10
    buffer_size: 100
    memory_size: 128

```

Slika 8: **Parametri nevronske mreže**

Tabela 2: **Razlaga parametrov [9]**

batch_size	Število izkušenj v vsaki ponovitvi gradientnega spuščanja. Ta mora biti vedno večkrat manjši od buffer_size. Če uporabljate neprekinjena dejanja, mora biti ta vrednost velika (približno 1000 sekund). Če uporabljate samo diskretna dejanja, mora biti ta vrednost manjša (približno 10 sekund).
buffer_size	PPO: število izkušenj, ki jih je treba zbrati pred posodobitvijo modela politike. Ustreza temu, koliko izkušenj je treba zbrati, preden se naučimo ali posodabljamo model. To bi moralo biti večkrat večje od batch_size. Običajno večji buffer_size ustreza stabilnejšim posodobitvam usposabljanja.
hidden_units	Število enot v skritih slojih nevronske mreže. Ustreza temu, koliko enot je v vsaki popolnoma povezani plasti nevronske mreže. Pri preprostih težavah, kjer je pravilno dejanje enostavna kombinacija opazovalnih vhodov, mora biti število majhno. Za težave, kjer je dejanje kompleksna interakcija med opazovanimi spremenljivkami, bi število moralo biti večje.
memory_size	Velikost pomnilnika, ki ga mora hraniti agent. Za uporabo LSTM usposabljanje zahteva zaporedje izkušenj namesto posameznih izkušenj. Ustreza velikosti množice števil, ki se uporabljajo za shranjevanje skritega stanja politike ponavljajoče se nevronske mreže. Ta vrednost mora biti večkratnik 2 in se mora prilagajati količinam informacij, za katere pričakujete, da si jih bo agent moral zapomniti, da lahko uspešno dokonča nalogo.
num_epoch	Število prehodov skozi medpomnilnik izkušenj pri izvajanju optimizacije gradientnega spuščanja. Večji, kot je batch_ size, bolj sprejemljivo je večje število tega parametra. Zmanjšanje tega bo zagotovilo stabilnejše posodobitve za ceno počasnejšega učenja.
num_layers	Število skritih plasti v nevronski mreži. Ustreza temu, koliko skritih plasti je prisotnih po vnosu opazovanja ali po CNN kodiranju vizualnega opazovanja. Za preproste težave bo manj plasti verjetno treniralo hitreje in učinkoviteje. Za kompleksnejše težave s krmiljenjem bo morda potrebnih več plasti.
sequence_length	Določa, kako dolga morajo biti zaporedja izkušenj med treningom. Upoštevajte, da če je ta številka premajhna, si agent ne bo mogel zapomniti stvari v daljšem časovnem obdobju. Če je to število preveliko, bo nevronska mreža potrebovala dlje, da se usposobi.

zmanjšanja čakalnih časov, izboljšanja pretočnosti prometa in povečanja splošne varnosti na cestah. Ta pristop predstavlja sodobno rešitev za izzive, s katerimi se srečujemo pri upravljanju urbanega prometa.

Na sliki 8 vidimo končne parametre za ustvarjanje naše nevronske mreže. Do prikazanih parametrov smo prišli tako, da smo vsako iteracijo nevronske mreže testirali v simulaciji in glede na rezultate smo izbrali optimalne parametre. Posamezni parametri so razloženi v tabeli

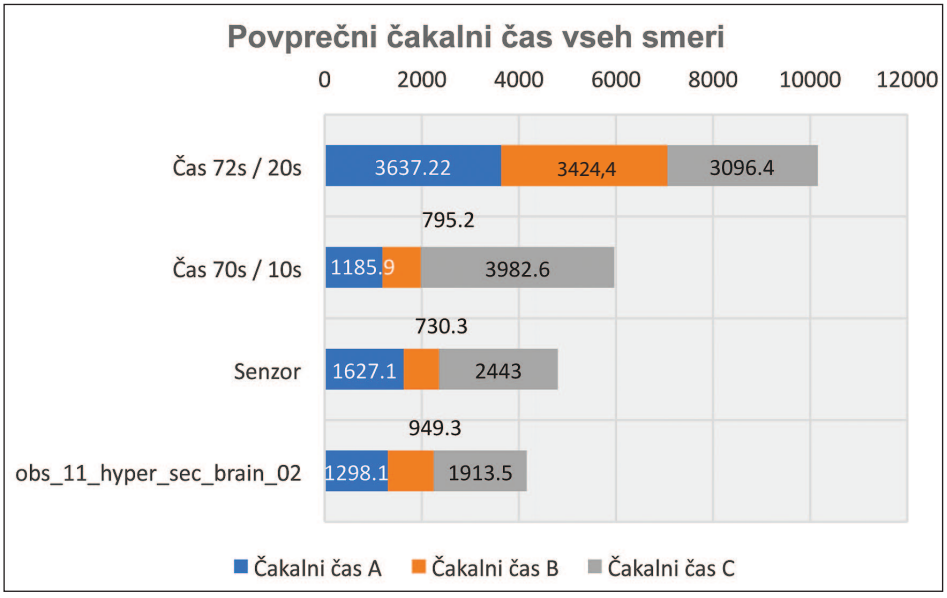
Ena komponenta učnih modelov s ogrodjem PyTorch je nastavitev vrednosti določenih atributov modela – hiperparametrov. Iskanje pravih vrednosti teh hiperparametrov lahko zahteva nekaj ponovitev. Posledično uporabljamo orodje za vizualizacijo, imenovano TensorBoard. Omogoča vizualizacijo določenih atributov agenta (na primer nagrada) skozi celotno usposabljanje, kar je lahko v pomoč tako pri gradnji intuicije za različne hiperparametre kot pri nastavljanju optimalnih vrednosti za vaše okolje. [7]

4 ANALIZA REZULTATOV OPTIMIZACIJE

Sedaj bomo pogledali rezultate vseh štirih stanj semaforiziranega križišča. Slika 9 ponuja vizualni pregled in primerjavo učinkov optimizacij na povprečne čakalne čase na križišču. Iz slike 9 so razvidne štiri vrstice, ki prikazujejo rezultate različnih stanj križišča:

1. Vrstica “Čas 72s / 20s” prikazuje izhodiščno stanje, kjer časi semaforjev niso bili spreminjani.
2. Vrstica “Čas 70s / 10s” odraža prvo optimizacijo, kjer smo spremenili čase semaforjev, kar je skrajšalo čakalne čase.
3. Vrstica “Senzor” prikazuje izboljšave, dosežene z uporabo senzorjev za dinamično prilagajanje časov semaforizacije glede na trenutne prometne razmere.
4. V zadnji vrstici, “obs_11_hyper_sec_brain_02”, so predstavljeni rezultati uporabe modela nevronske mreže za nadzor semaforiziranega križišča.

Ti rezultati potrjujejo učinkovitost sodobnih tehnologij in pristopov k upravljanju prometa na semaforiziranih križiščih. Optimizacija s pomočjo naprednih algoritmov strojnega učenja, kot je nevronska mreža, ponuja obetavne rešitve za izboljšanje prometne infrastrukture v urbanem okolju. Uporaba takšnih tehnologij ne samo izboljšuje pretočnost prometa in zmanjšuje čakalne čase, temveč tudi prispeva k večji varnosti, zmanjšanju stresa za voznike in splošnemu izboljšanju kakovosti bivanja v mestih.



Slika 9: **Rezultati simulacij vseh različnih stanj simulacije**

5 ZAKLJUČEK

V raziskavi smo optimizirali povprečni čakalni čas na izbranem semaforiziranem križišču. S pomočjo simulacijskega modela in strojnega učenja smo prišli do odgovora, na kakšen način in v kolikšni meri je mogoče optimizirati semaforizirano križišče. V zaključku naše raziskave je pomembno poudariti, da smo se med razvojem simulacijskega modela soočali z določenimi omejitvami. Osredotočili smo se izključno na analizo prometa na specifičnem križišču in v zelo omejenem časovnem okviru – za en dan in določeno uro. Kljub temu, da so rezultati pokazali značilne vzorce in omogočili določene optimizacije, moramo priznati, da je resnični svet bistveno bolj zapleten. Na prometne razmere nenehno vplivajo različni dejavniki, ki jih naš model trenutno še ne upošteva oziroma jih ne upošteva v celoti.

Analiza je pokazala, da so vse tri metode optimizacije znatno izboljšale pretočnost prometa na križišču:

- s prilagoditvijo časov semaforjev (Čas 70s / 10s) smo dosegli zmanjšanje skupnega čakalnega časa za približno 41 %,
- implementacija senzorjev, ki omogočajo prilagajanje semaforjev v realnem času, je zmanjšala čakalne čase za približno 53 %,
- najimpresivnejši rezultat je bil dosežen z uporabo nevronske mreže, ki je skupne čakalne čase zmanjšala za približno 59 %.

Zavedamo se, da so prometni sistemi dinamični in medsebojno povezani, zato je nujno, da naša naslednja faza razvoja vključuje razširitev simulacijskega modela na mrežo križišč. Vsako križišče v mestni mreži vpliva na delovanje sosednjih križišč, kar pomeni, da je treba za boljše razumevanje in optimizacijo prometnih tokov razviti model, ki lahko simulira in analizira več križišč hkrati. Tovrsten pristop bi nam omogočil bolj celostno razumevanje prometnih dinamik in vzorcev, kar bi posledično vodilo do učinkovitejših rešitev za zmanjšanje zastojev in izboljšanje pretočnosti.

Dolgoročno gledano je naš cilj razviti robustnejše in prilagodljive sisteme, ki se ne samo odzivajo na trenutne pogoje, ampak lahko predvidevajo in se proaktivno prilagajajo prihajajočim spremembam. To bo zahtevalo nadaljnje raziskave in razvoj v smeri vključitve naprednih tehnologij, kot sta umetna inteligenca in strojno učenje, ki lahko simulacijam prometa dodajo nove razsežnosti inteligence in predikativne moči.

V sorodni raziskavi je Wiering napisal:

V prihodnjem delu bi radi preizkusili svoje sisteme na bolj realističnih simulatorjih prometa, v katere želimo dodati tudi javni prevoz, ki bi moral imeti prednost, saj ima več potnikov. [1]

Prometne sisteme je mogoče optimizirati na različne načine in naša ratiskava je samo temelj za nadaljnje študije in razvoj na področju prometnega

inženiringa, s končnim ciljem ustvarjanja bolj odzivnih in inteligentnejših prometnih sistemov, ki bodo pripomogli k boljši mobilnosti, varnosti in kakovosti življenja v urbanem okolju.

6 LITERATURA

[1.] Wiering, M. (2000). Multi-Agent Reinforcement Learning for Traffic Light Control. University of Utrecht, Department of Computer Science

[2.] Wiering, M., van Veenen, J., Vreeken, J., & Koopman, A. (2004). Intelligent Traffic Light Control. Utrecht: institute of information and computing sciences, utrecht university.

[3.] Mohammad Noaeen, Atharva Naik, Liana Goodman, Jared Crebo, Taimoor Abrar, Zahra Shakeri Hossein Abad, Ana LC Bazzan, Behrouz Far (2022/8/1). Reinforcement learning in urban network traffic signal control: A systematic literature review. Expert Systems with Applications.

[4.] Tuliper, A. (1. Avgust 2014). Unity : Developing Your First Game with Unity and C#. MSDN Magazine, str. 36-37.

[5.] Unity Technologies. (26. Oktober 2020). Unity Documentation. Pridobljeno iz unity3d: https://docs.unity3d.com/Manual/index.html

[6.] Hejlsberg, A., Torgersen, M., Wiltamuth, S., & Golde, P. (2011). The C# Programming Language Fourth Edition. Boston: Addison-Wesley.

[7.] Unity Technologies. (5. Februar 2022). github.com/Unity-Technologies/ml-agents. Pridobljeno iz github.com/Unity-Technologies: https://github.com/Unity-Technologies/ml-agents

[8.] Géron, A. (2019). Hands-On Machine Learning with Scikit--Learn, Keras, and TensorFlow, 2nd Edition. Newton: O'Reilly Media, Inc.

[9.] Unity Technologies. (5. November 2021). Unity MLAgents Documentation. Pridobljeno iz ML Agents: https://docs.unity3d.com/Manual/com.unity.ml-agents.html

[10.] Blender. (4. Februar 2022). About - blender.org. Pridobljeno iz blender.org: https://www.blender.org/about/

▪

Šemso Hrnjičić je razvijalec programske opreme in svetovalec na področju preiskav letalskih nesreč in incidentov na Ministrstvu za obrambo Republike Slovenije. Trenutno deluje kot razvijalec v podjetju Endava, kjer sodeluje pri razvoju naprednih rešitev na področju mikroskopov za mednarodno podjetje. Njegova raziskovalna in strokovna področja vključujejo razvoj programske opreme, pametne sisteme in avtomatizacijo, simulacije ter uporabo umetne inteligence pri optimizaciji procesov. Posebej ga zanimajo aplikacije strojnega učenja, 3D simulacij in vizualizacij pri reševanju kompleksnih tehničnih in organizacijskih izzivov.

▪

Dr. Uroš Rajkovič je izredni profesor s področja informacijskih sistemov na Fakulteti za organizacijske vede Univerze v Mariboru. Njegova raziskovalna področja vključujejo umetno inteligenco in informacijske sisteme za podporo odločanju, s poudarkom na večkriterijskem modeliranju in optimizaciji procesov. Posebej ga zanimajo uporabe metod ekspertnih sistemov, simulacij ter naprednih algoritmov pri reševanju kompleksnih organizacijskih, tehničnih in družbenih problemov.