Univerzitet u Beogradu

Matematički fakultet

Master rad

Kontrola autonomnog vozila u virtuelnom saobraćajnom okruženju

Autor:
Dejan Mitrović

Mentor:
prof. Predrag Janičić

Beograd, 2015.

Sadržaj

1. Uvod

Od prvih automobila do danas, automobilska industrija se razvija velikom brzinom. Automobili su danas snažniji, brži i ima ih sve više. Negativna strana tih promena je to što vozači često nisu u stanju da adekvatno reaguju na okolnosti u saobraćaju. Bezbednost u saobraćaju je stalni problem. Samo u Srbiji, u saobraćajnim nesrećama dnevno u proseku poginu dve osobe [1].

U težnji da se bezbednost učesnika u saobraćaju podigne na najviši mogući nivo, sve više se u automobile ugrađuju napredne tehnologije za pomoč vozaču. Automatsko održavanje brzine, automatsko održavanje rastojanja, prepoznavanje saobraćajnih znakova i samostalno parkiranje su prvi koraci ka budućim samonavodećim automobilima, dok su kompletni sistemi za automatsku vožnju već u razvoju od strane mnogih vodećih proizvođaća automobila. [2, 3, 4, 5, 6]

Postoje dva osnovna koncepta pametnih automobila. Prvi je koncept pametnih automobila zasnovan na pametnim putevima. Osnovna ideja je korišćenje posebnih puteva koji šalju podatke automobilima na njima kako bi oni mogli da prate put. Iako je on tehnički jednostavniji, negativna strana ovog pristupa je što ograničava kretanje autonomnim automobilima samo na područja pokrivena odgovarajućim putevima.

Drugi pristup se zasniva na pametnim automobilima koji sve podatke potrebne za odlučivanje dobavljaju samostalno. Ovaj pristup je posebno dobio na popularnosti nakon DARPA Urban Challenge takmičenja 2007. godine. DARPA (agencija Ministarstva odbrane SAD za razvoj novih tehnologija) organizuje takmičenja za autonomna terenska vozila još od 2004. godine, a 2007. je po prvi put organizovala takmičenje za autonomna vozila u urbanom okruženju [7].

Poslednjih godina veliku medijsku pažnju privukla je kompanija Google i njen sistem za autonomnu vožnju. Na čelu Google tima za razvoj autonomnih automobila je Sebastian Thrun, koji je sa svojim timom sa Stenford Univerziteta osvojio DARPA Grand Challenge 2005. i osvojio drugo mesto na DARPA Urban Challenge 2007. Autonomni automobili kompanije Google prešli su preko milion kilometara bez incidenata izazvanih sistemom za automatsku vožnju [8].

2. Model virtuelnog saobraćajnog okruženja

Model virtuelnog saobraćajnog okruženja se bazira na stvarnim pravilima saobraćaja, uz neka pojednostavljenja. Na primer, svi putevi su u ravni, nema slepih puteva i slično. Ponašanje učesnika u saobraćaju (automobila i pešaka) je takođe pojednostavljeno: na primer, pešaci se uglavnom kreću pravolinijski.

2.1 Osnovne komponente modela

Agent

Agent je centralni objekat u modelu. To je automobil koji je u stanju da se kreće samostalno, bez ljudskog vozača ili spoljašnjih instrukcija. U konkretnoj implementaciji svi automobili koriste iste algoritme za kretanje, ali agentom ćemo zvati jedan konkretan automobil koji se prati i razmatra.

Automobil

Automobil je učesnik u saobraćaju koji se kreće po putevima. Pretpostavlja se da svi automobili poštuju ista saobraćajna pravila.

Put

Automobili se kreću po putevima. Svaki put sastoji se od dve jednosmerne trake. Mogućnost preticanja automobila korišćenjem paralelne trake nije obuhvaćena modelom. Svi putevi imaju istu širinu i na sebi mogu imati pešačke prelaze.

Niz puteva koji spajaju agenta sa njegovom destinacijom se naziva "putanja" agenta.

Raskrsnica

Raskrsnice su ukrštanja tri ili više puteva. U raskrsnici se može ukrštati proizvoljan broj puteva pod proizvoljnim uglovima. Prvenstvo prolaza kroz raskrsnicu je određeno semaforima na svakom putu, a pored toga koristi se i "pravilo desne strane". Svaki put u raskrsnici ima pešački prelaz koji omogućava pešacima prelazak preko raskrsnice.

Raskrsnice takođe služe i kao početne i krajnje tačke za traženje najkraćeg puta.

U raskrsnicu, odnosno iz nje, može voditi proizvoljan broj puteva. Za potrebe određivanja putanje agenta unutar raskrsnice se nalaze kratki putevi koji spajaju ulazne i izlazne puteve. Ti putevi se nazivaju "unutrašnji putevi". Putevi između raskrsnica nazivaju se i "spoljašnji" putevi. Prilikom određivanja putanje agenta, određuju se spoljašnji putevi do raskrsnice i od nje, a u putanju se dodaje i unutrašnji put koji ih spaja. Nakon određivanja putanje agent neće koristiti ostatak raskrsnice već će samo pratiti unutrašnji put kroz nju.

Mreža puteva

Mreža puteva je skup svih puteva i raskrsnica. Pretpostavlja se da su sve raskrsnice povezane putevima, to jest da između svake dve rasksnice postoji putanja (niz puteva i raskrsnica koji ih spaja). Agent ima pristup konfiguraciji cele mreže puteva.

Mreža puteva uključuje sledeće informacije:

- broj raskrsnica u mreži;
- informacije o svakoj raskrsnici;
- broj puteva koji se ukrštaju u raskrsnici;
- raskrsnice do kojih ti putevi vode;
- koordinate svih ključnih tačaka puteva;
- početna stanja svih semafora;
- informacije o eventualnim pešačkim prelazima na putevima;
- informacije o "unutrašnjim putevima" u raskrsnicama

Semafor

Semafori se nalaze na raskrsnicama i kontrolišu prvenstvo prolaza učesnika u saobraćaju – automobila i pešaka.

Pešak

Pešaci su učesnici u saobraćaju koji se ne kreću duž puteva već van njih. Oni u toku kretanja mogu prelaziti preko puteva, na pešačkim prelazima ili van njih.

Pešački prelaz

Pešački prelaz je mesto na putu namenjeno za prelazak pešaka. Iako pešaci mogu da prelaze puteve i van pešačkih prelaza, pešački prelazi su prvi izbor. Prelazi se nalaze i na raskrsnicama kao jedini način za prelazak pešaka preko raskrsnice.

2.2 Funkcionalnosti agenta

Agenti u ovom modelu moraju da imaju sledeće funkcionalnosti:

- Određivanje najkraće putanje između početne pozicije (raskrsnice) i zadate raskrsnice;
- Kretanje po putu optimalnom brzinom;
- Prelazak na sledeći odgovarajući put u raskrsnici;
- Održavanje odstojanja u odnosu na automobile ispred njega na istom putu;
- Razmatranje prvenstva prolaza u raskrsnicama i generalno izbegavanje sudara sa drugim automobilima u okolini;
- Poštovanje semafora i pravila desne strane u raskrsnicama;
- Izbegavanje sudara sa pešacima koji prelaze put, odnosno stajanje ispred pešaka koji se spremaju da pređu put;
- Dolazak od početne tačke do zadate destinacije prateći sve navedene uslove.

2.3 Učesnici u saobraćaju

Sa tačke gledišta agenta, učesnici u saobraćaju su svi objekti koji mogu uticati na kretanje agenta: drugi automobili, pešaci, semafori i pešacki prelazi.

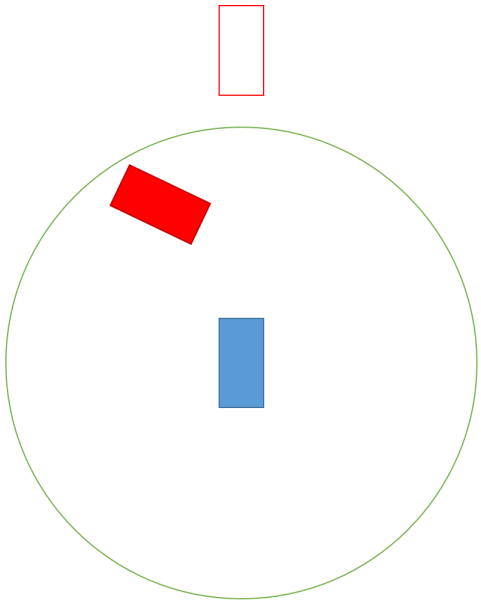
2.3.1 Vidno polje agenta

Za razliku od mreže puteva o kojoj ima potpune informacije, agent nema potpune informacije o drugim učesnicima u saobraćaju. Agent ima informacije samo o učesnicima u saobraćaju u svojoj neposrednoj okolini, i to samo vidljive informacije: poziciju, pravac kretanja i trenutnu brzinu. (slika 2.1)

2.3.2 Semafori

Semafori kontrolišu prvenstvo prolaza u raskrsnicama. U svakodnevnom saobraćaju, semafori prolaze kroz ciklus od 5 različitih stanja: crveno svetlo, crveno i žuto svetlo, zeleno svetlo, trepćuće zeleno svetlo, žuto svetlo. (slika 2.2)

Crveno i zeleno svetlo označavaju dva različita stanja prolaznosti puta. "Crveno i žuto svetlo", kao i "trepćuće zeleno" ne predstavljaju promenu stanja, već su stanja koja postoje kako bi se vozači unapred obavestili o predstojećoj promeni stanja, kako bi mogli da obrate pažnju na semafor i pripreme se za polazak, odnosno zaustavljanje. Za autonomne agente takva priprema nije potrebna, jer se pretpostavlja da oni uvek obraćaju pažnju na promenu semafora i mogu momentalno da reaguju na promenu, tako da model ne uključuje



Slika 2.1: Ilustracija vidnog polja agenta

ova dva stanja. S druge strane, žuto svetlo na semaforu omogućava vozilima da na vreme reaguju na predstojeće crveno svetlo. Kao takvo, neophodno je i autonomnim agentima.

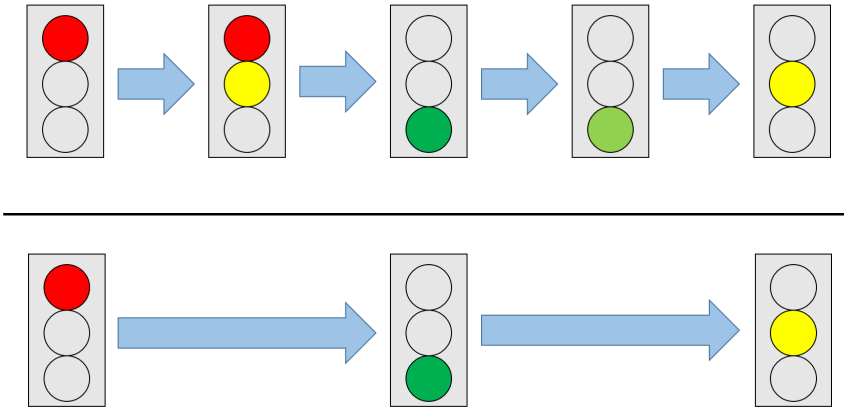
2.3.3 Drugi automobili

Svi automobili u modelu se ponašaju na isti način i prate ista saobraćajna pravila. Ona uključ uju praćenje puta, održanje rastojanja između automobila, poštovanje prvenstva prolaza u raskrsnicama (uključujuci poštovanje semafora i pravila desne strane), kao i ustupanje prvenstva prolaza pešacima.

Mogućnosti automobila određene su sledećim parametrima:

- maksimalna brzina;
- ubrzanje;
- sila kočenja (maksimalno usporenje);
- maksimalni ugao skretanja u jedinici vremena;
- domet vidnog polja.

Od navedenih parametara zavisi kretanje agenta i od njih zavise sva izračunavanja u toku kretanja.



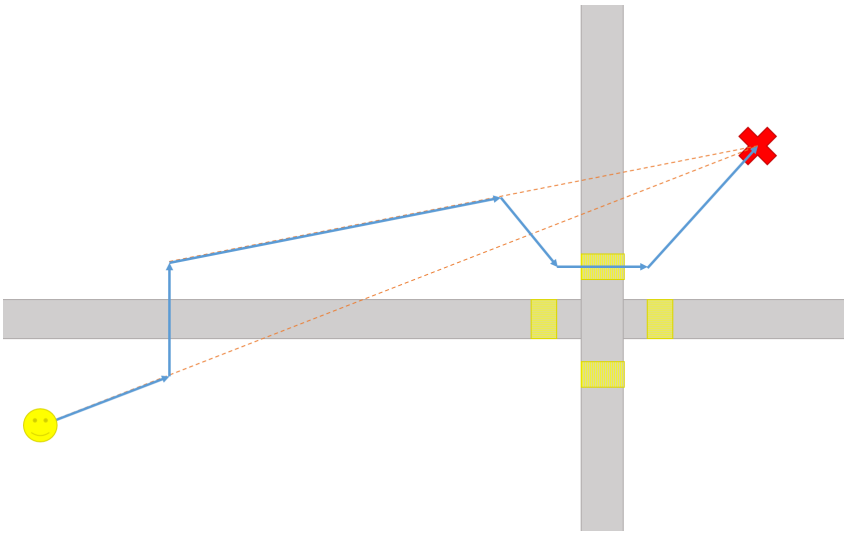
Slika 2.2: Različita stanja semafora u saobraćaju (gore) i u modelu (dole)

2.3.4 Pešaci i pešački prelazi

Pešaci su učesnici u saobraćaju koji se generalno kreću van puteva i prelaze ih po potrebi.

Pešaci se kreću ka proizvoljnim pozicijama u okviru mreže puteva. Ukoliko se u toku kretanja približe putu, pokušaće da ga pređu. Najpre će potražiti pešački prelaz u blizini i preći će put preko njega, ako ga nađu. U suprotnom će, posle kraćeg čekanja, preći put najkraćim putem, normalno na put. (slika 2.3)

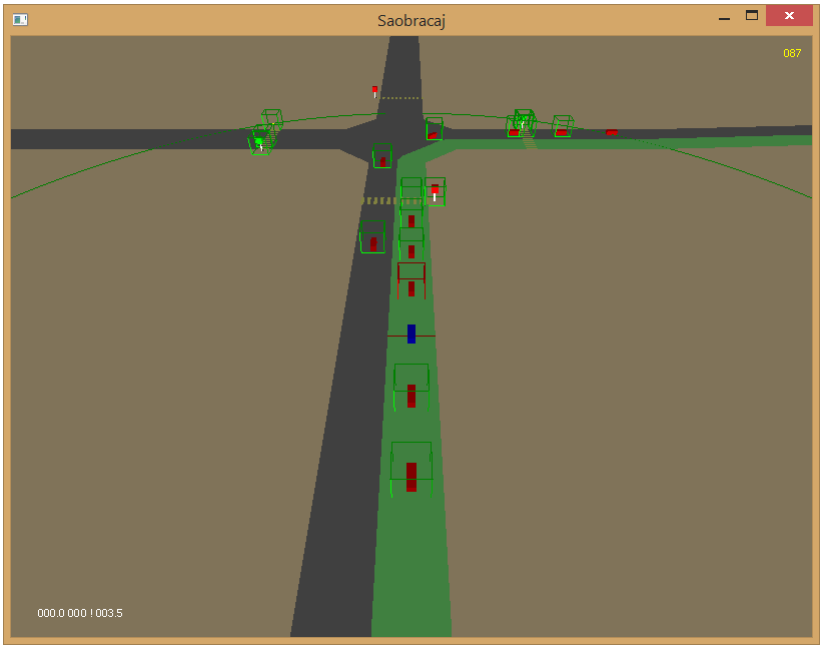
Na raskrsnicama, mogućnost prelaska preko pešačkih prelaza zavisi od stanja semafora. Pešaci će prelaziti samo preko pešačkih prelaza na putevima sa semaforima zatvorenim za automobile (crveno svetlo za automobile na putu).



Slika 2.3: Primer putanje pešaka

3. Implementacija

Model saobraćaja je implementiran kao C++ aplikacija. U njoj se simulira kretanje određenog broja autonomnih automobila i pešaka na datoj mreži puteva. Automobili i pešaci su međusobno nezavisni. Svi automobili u aplikaciji funkcionišu kao autonomni agenti, a aplikacija prati ponašanje jednog od njih. Početna i krajnja pozicija agenta se zadaju neposredno nakon pokretanja aplikacije, dok se ostali automobili nasumično kreću po mreži puteva. Broj agenata i pešaka, kao i parametri agenta, se zadaju preko konfiguracionih datoteka, dok se parametri ostalih automobila i destinacije pešaka nasumično biraju. Aplikacija vizuelno prati agenta kroz pogled iz trećeg lica ili iz 2D ptičje perspektive.



Slika 3.1: Izgled aplikacije

3.1 Korišćeni algoritmi i biblioteke

Svi potrebni algoritmi su razvijeni i implementirani samostalno, tako da aplikacija koristi vrlo mali broj postojećih biblioteka. Izuzeci su deo za grafički prikaz aplikacije, koji koristi OpenGL, i nekoliko poznatih algoritama i koncepata kao što je algoritam A*, koji su implementirani na osnovu postojećih specifikacija.

3.1.1 OpenGL

Izuzetak je deo za grafički prikaz aplikacije, za koji je korišćen OpenGL i freeGLUT biblioteka. OpenGL je višeplatformski programski interfejs koji je de facto standard za grafičke simulacije.[10]

Prva verzija OpenGL-a je objavljena 1997. godine, i podržavala je fiksni proces iscrtavanja. Nakon toga objavljeno je nekoliko novijih verzija OpenGL-a koje su donele nove funkcionalnosti, uključujući programabilni proces iscrtavanja. Programabilni proces iscrtavanja je znatno fleksibilniji i moćniji, ali je komplikovaniji za jednostavne projekte. Fiksni proces iscrtavanja je i dalje podržan u novim verzijama OpenGL-a, uglavnom zbog kompatibilnosti sa starim kodom. Aplikacija je pravljena korišćenjem fiksnog procesa iscrtavanja, koji je bio dovoljan za potrebe projekta.

3.1.2 GLUT

GLUT (OpenGL Utility Toolkit) je biblioteka koja olakšava rad sa OpenGL-om. GLUT podržava:

- pravljenje prozora i upravljanje njima
- procesiranja ulaza sa tastature i miša
- procesiranje bazirano na događajima
- rad sa tajmerima
- funkcije za iscrtavanje osnovnih geometrijskih figura
- rad sa fontovima
- rad sa kontekstnim menijima

Biblioteka GLUT je prestala sa razvojem sa verzijom 3.7, 1998. godine. FreeGLUT je biblioteka otvorenog koda koja je naslednik GLUT-a.[11] Ona je unazad kompatibilna sa GLUT aplikacijama i može se koristiti umesto GLUT-a. FreeGlut (i GLUT) je neazvisna od operativnog sistema i omogućava prenosivost koda.

3.1.3 Algoritam A*

Kao algoritam za izračunavanje najkraćeg puta za autonomna vozila korišćen je A* algoritam. A* je efikasan algoritam za traženje najkraćeg puta u grafu, baziran na Dejkstrinom algoritmu.[9] Dejkstrin algoritam pretrazuje graf obilaskom čvorova i čuvanjem trenutne minimalne cene puta do svakog čvora. Redosled obrade čvorova je na osnovu najmanje trenutne cene puta od početnog do krajnjeg čvora. Nakon izvršavanja algoritma moguće je i rekonstruisati najkraći put. A* poboljšava brzinu pretrage korišćenjem heuristike, što dovodi do obrade manjeg broja čvorova.

Algoritam A* smešta čvorove koje posećuje u dve liste: otvorenu listu, koja sadrži čvorove koji su posećeni ali još uvek nisu obrađeni, i zatvorenu listu, koja sadrži čvorove koji su potpuno obrađeni i na koje se neće vraćati. U svakom koraku algoritam bira jedan čvor iz otvorene liste, obrađuje ga i prebacuje u zatvorenu listu. Ono što poboljšava performanse algoritma u odnosu na Dejkstrin algoritam je način izbora čvorova iz otvorene liste. Oni se biraju na osnovu funkcije evaluacije

$$f(x) = g(x) + h(x)$$

pri čemu je $g(x)$ trenutna cena puta do tog čvora, a $h(x)$ vrednost puta od tog čvora do ciljnog procenjena heuristikom. Ukoliko je za heuristiku uzme konzistentna funkcija (takva da za svaka dva susedna čvora u i v važi da je $h(u) \leq h(v) + c(u, v)$, gde je $c(u, v)$ cena grane između u i v), algoritam A* je optimalan.

U aplikaciji A* se koristi za određivanje putanja automobila. Graf po kojem se vrši pretraga je mreža puteva, pri čemu se raskrsnice smatraju čvorovima a putevi između njih granama. Kao heuristika korisceno je direktno rastojanje između raskrsnica, što je konzistetna heuristika.

3.1.4 Konačni automati

Za modelovanje ponašanja pešaka kao dodatnih učesnika u saobraćaju korišćen je jednostavan konačni automat. Konačni automati su apstraktne mašine koje se sastoje od konačnog broja stanja. Automat se nalazi u jednom stanju, koje se naziva "trenutno stanje". Moguće je preći iz trenuntog stanja u neko drugo na osnovu ulaza. Pravila prelaska iz stanja u stanje zavise od trenutnog stanja i vrednosti ulaza.

U aplikaciji kretanje pešaka je modelovano konačnim automatom, gde su stanja pešaka kao što su "kretanje" i "stajanje na pešačkom prelazu", dok su pravila za prelazak iz stanja u stanje komplikovanija i zavise od okoline u kojoj se nalazi pešak.

3.1.5 Simulacija kretanja u realnom vremenu

Simulacija kretanja agenata se obavlja deljenjem na diskretne vremenske intervale fiksne dužine (eng. fixed time step simulation). Sva izračunavanja

i fizičke formule se diskretizuju i primenjuju se u fiksnoj jedinici vremena (u jednom diskretnom vremenskom intervalu). U slučaju da izračunavanje između dva vremenska intervala traje duže dolazi do usporavanja simulacije, ali je krajnji rezultat i dalje korektan. Aplikacija koristi trajanje vremenskog intervala od 20 ms (50 Hz), što se pokazalo kao dovoljno dobro za simulaciju kretanja u realnom vremenu.

3.2 Mreža puteva

Mreža puteva služi kako za čuvanje informacija o svim putevima, tako i za izračunavanje najkraćeg puta i čuvanje informacija o trenutnoj putanji.

Podaci o mreži se učitavaju iz tekstualne datoteke. U njoj su sekvencijalno zapisane koordinate raskrsnica i puteva, kao i semafora i pešačkih prelaza, i druge brojne vrednosti potrebne za rekonstrukciju mreže puteva, kao što su broj puteva u raskrsnicama. Svaki agent ima svoju kopiju mreže, koja se koristi i za čuvanje izračunate putanje i praćenje napretka po njoj.

3.2.1 Semafori

Prvenstvo prolaza u svakoj raskrsnici je određeno semaforima. Početna stanja semafora (koji su otvoreni a koji su zatvoreni) se učitavaju kao parametri mreže puteva. Svi semafori u mreži puteva se menjaju u isto vreme. Na taj način se očuvava sinhronizovanost semafora u raskrsnicama. Iako su semafori delovi mreže puteva, nema svaka mreža svoju kopiju već postoji samo po jedna instanca svakog semafora koja je dostupna svim agentima.

3.3 Implementacija pešaka

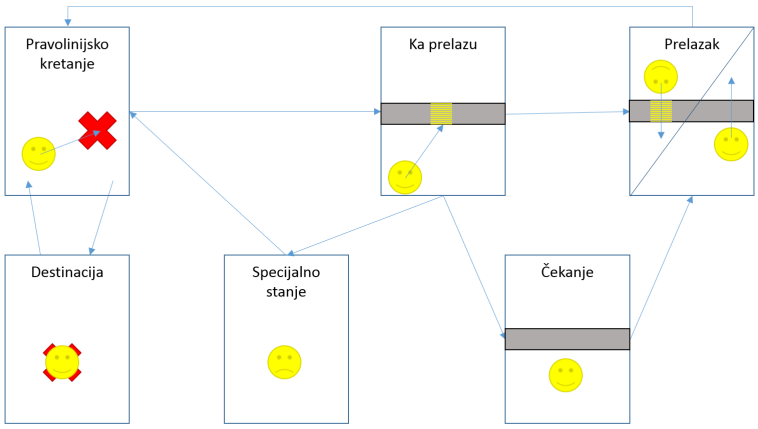
Za potrebe aplikacije, postoji relativno mali broj pešaka koji se konstantno kreću ka nasumičnim mestima na mapi. Ovo je dovoljno da simulira nasumičnost pešaka u saobraćaju.

Pešaci su implementirani pomoću jednostavnog konačnog automata sa šest stanja (slika 3.2). Njime je pokriveno kretanje pešaka i akcije prelaženja puteva.

Pravolinijsko kretanje

Svaki pešak se kreće ka nasumično odabranoj tački unutar mreže puteva (njegova "destinacija"). Sve dok se ne nalazi blizu nekog puta, pešak se podrazumevano kreće direktno ka destinaciji.

Na početku kretanja, na osnovu pravca od početne pozicije pešaka do destinacije izdvaja se podskup puteva koje pešak može preseći u toku kretanja. U toku kretanja, kada se proverava da li je pešak u blizini puta, proveravaju se samo izdvojeni putevi, radi efikasnosti. Lista izdvojenih puteva se ponovo izračunava kada pešak promeni pravac, na primer nakon prelaska preko pešačkog prelaza.



Slika 3.2: Dijagram stanja pešaka

Kada u toku kretanja pešak dođe u neposrednu blizinu puta, pravac kretanja se menja: analiziraju se pešački prelazi na tom putu i pešak se usmerava prema onom od njih koji će ga dovesti bliže destinaciji. U izbor ulaze svi pešački prelazi na putu pored kojeg se pešak nalazi. Ukoliko nijedan pešački prelaz nije odgovarajući ili nema pešačkih prelaza u okolini, pešak će za mesto prelaska izabrati najbližu tačku na putu.

Kada se pešak približi raskrsnici, bira se i pešački prelaz (ili više njih) preko kojih će da pređe. U ovoj situaciji kandidati su svi prelazi na raskrsnici. Pešak će izabrati one prelaze koji ga vode u isti deo ravni sa njegovom destinacijom.

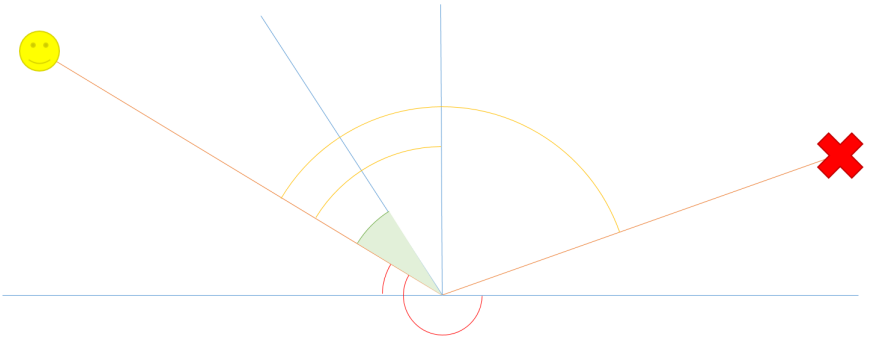
Pozicija raskrsnice se razmatra kao koordinatni početak koordinatnog sistema. Izračunavaju se uglovi između koordinatnog početka, pravca ka destinaciji i svakog od puteva u raskrsnici, kao i ugao do pravca trenutne pozicije pešaka. Za pešački prelaz se bira prelaz preko onog puta koji je između pešaka i destinacije, a najbliži pešaku. (slika 3.3)

Kretanje ka prelazu

Nakon što je izabran pešački prelaz, pešak će promeniti pravac kretanja i uputiti se ka istom. Pešak će se kretati ka bližem kraju prelaza. Jednom kada stigne do njega, ukoliko prelaz nema semafor ili je semafor otvoren, pristupiće prelasku puta.

Čekanje

Ukoliko pešak prelazi put van pešačkog prelaza, on će se najpre zaustaviti i sačekati pre nego što počne da prelazi put. Zatim će preći put. Prelazak preko pešačkih prelaza ne zahteva čekanje, osim ako je u pitanju pešački



Slika 3.3: Ilustracija algoritma za izbor prelaza pri prolasku kroz raskrsnicu

prelaz sa otvorenim semaforom za automobile. Pešak neće prelaziti put ukoliko se direktno ispred njega nalazi automobil, na primer ako se radi o redu automobila koji čekaju zbog nekog ispred. Ovo važi kako za prelaske preko pešačkih prelaza, tako i za direktan prelazak preko puta.

Prelazak puta

Prelazak preko puta se obavlja najkraćim putem, normalno na pravac puta. Nakon prelaska, pešak ponovo nastavlja kretanje ka destinaciji. Pošto odstupanje sa pravca radi prelaska puta može da dovede do znatne promene pravca ka destinaciji, nakon prelaska se vrši ponovno izračunavanje izdvojenih puteva, to jest puteva na koje pešak može naići u toku ostatka kretanja.

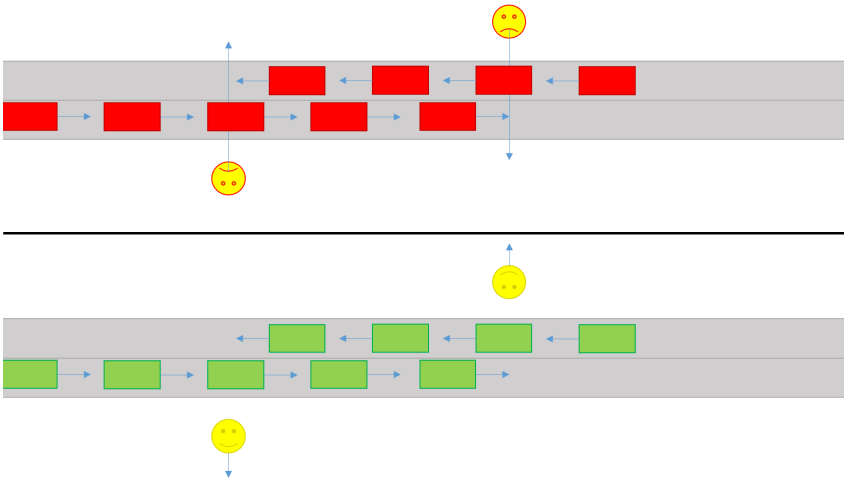
Dolazak na destinaciju

Destinacija je uvek tačka van puteva i raskrsnica. Jednom kada pešak stigne na destinaciju, bira novu proizvoljnu tačku kao sledeću. U isto vreme se vrši i određivanje izdvojenih puteva koje će pešak preseći. Određivanje izdvojenih puteva se vrši prolaskom kroz sve puteve, ali to ne utiče bitno na performanse sistema.

Specijalno stanje

Ovo stanje rešava par specifičnih situacija koje se javljaju u toku kretanja pešaka, a koje nisu obuhvaćene ostalim stanjima. Te situacije mogu izazvati zaglavljivanje pešaka i/ili automobila i moraju se rešiti na na poseban način.

Jedna od takvih situacija je je lančano čekanje pešaka na automobile i obratno. Ovakva situacija je moguća usled jednostavnog ponašanja pešaka. Pešaci koji čekaju na prelazak puta neće ga preći ukoliko se ispred njih nalazi automobil. Isto tako, automobili koji čekaju pešake (ili druge automobile) ispred sebe neće nastaviti da se kreću dok pešaci ne pređu put.



Slika 3.4: Lančano blokiranje pešaka i automobila (gore) i rešenje okretanjem pešaka (dole)

Za rešavanje ovakvih situacija, pešaci koji čekaju na prelazak duže od određenog vremena će promeniti smer kretanja za 180°. Na taj način će odblokirati automobile koji ih čekaju i potencijalno razrešiti lanac čekanja (slika 3.4). Ovaj postupak se može tumačiti kao signaliziranje pešaka automobilima da mogu da prođu ispred njih.

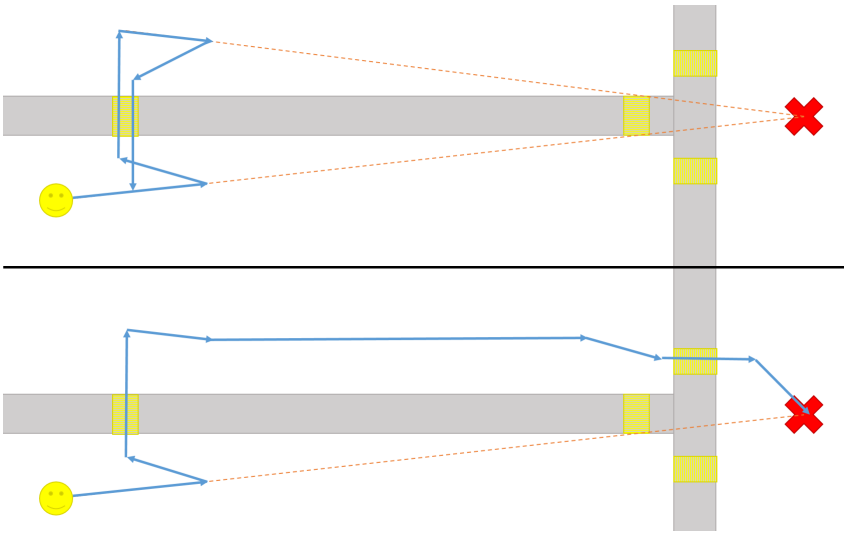
Ovo stanje se aktivira i ukoliko pešak pokuša da pređe isti pešački prelaz dva puta zaredom. Ova situacija se javlja u slučaju da je destinacija pešaka na takvoj poziciji (obično na pravcu puta koji se prelazi) da svaki prelazak puta navodi pešaka da mu se ponovo približi (slika 3.5).

Kao rešenje ovakve situacije, kada pešak pokuša da krene ka istom pešačkom prelazu više od jednom zaredom, on će promeniti pravac kretanja i kretaće se paralelno sa putem, u pravcu destinacije. Kratko kretanje na ovaj način je dovoljno da se jednakost daljina promeni i pešak se zatim vraća na normalno kretanje.

3.4 Ponašanje agenta

Ponašanje agenta je implementirano na tri, uglavnom nezavisna, nivoa:

- Na najvišem nivou je planiranje putanje, to jest određivanje najkraćeg puta na osnovu poznate mreže puteva.
- Logika vezana za praćenje puta, izračunavanje uglova skretanja i prelazak na sledeći put na putanji. Ovde spada i interakcija sa ostalim učesnicima



Slika 3.5: Ilustracija situacije kada pešak pređe isti prelaz više puta (gore) i rešenje (dole)

u saobraćaju unutar vidnog polja agenta i predviđanje onih van njega.

- Na najnižem nivou je pojednostavljeni fizički model kretanja.

3.4.1 Planiranje putanje

Za planiranje putanje, mreža puteva se razmatra kao graf. Raskrsnice su čvorovi grafa, a putevi između njih su grane grafa. Za određivanje najkraćeg puta primenjuje se A* algoritam za traženje najkraćeg puta u grafu. Za dužinu grane koristi se dužina puta između raskrsnica, dok se za heuristiku koristi rastojanje vazdušnom linijom.

Nakon primene algoritma, formira se putanja - lista povezanih puteva kojima agent treba da se kreće. Agent u toku kretanja više ne koristi opis mreže već samo prati puteve iz putanje jedan za drugim.

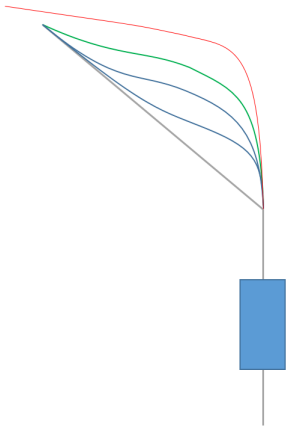
Za početno izračunavanje putanje (pri kreiranju automobila) aplikacija orijentiše automobile u pravcu prvog puta na putanji. Za kasnija izračunavanja putanje, kada je agent već u raskrsnici u vreme kada se vrši izračunavanje, put na kojem se trenutno nalazi neće biti uključen u izračunavanje najkraćeg puta, jer polukružno okretanje nije moguće. Ovaj princip koriste svi automobili u aplikaciji da oponašaju kontinualno kretanje.

3.4.2 Praćenje puta i skretanje

Putanja koju agent dobija spaja puteve u niz. Kako se svi putevi sastoje od nizova ključnih tačaka, putanja se može razmatrati kao jedan veliki put.

Praćenje putanje se onda svodi na praćenje ključnih tačaka. Kada su ključne tačke približno kolinearne, praćenje puta je trivijalno. Komplikovaniji slučaj je skretanje u krivinama.

Za model kretanja koji agenti koriste, najveći problem u toku praćenja puta je određivanje optimalne brzine kretanja. Pri prevelikoj brzini, može se desiti da agent nema dovoljno vremena da ispravi pravac ka sledećoj tački i da izleti sa puta. (slika 3.6) Na osnovu karakteristika agenta moguće je odrediti maksimalnu ulaznu brzinu u segment puta tako da agent ne izleti sa puta u toku tog segmenta. Međutim, kretanje tako izračunatom maksimalnom brzinom može dovesti do toga da ulazna brzina u sledeći segment puta bude prevelika, što dovodi do izletanja sa puta. Pošto bezbedna optimalna brzina za segment puta uvek zavisi od sledećeg segmenta, jedini način za izračunavanje optimalnih brzina je računanje za celu putanju unapred, od poslednjeg segmenta poslednjeg puta unazad. Negativna strana ovog pristupa je što zahteva potpunu preciznost u toku kretanja; i najmanje odstupanje bi poništilo sva izračunavanja. Zbog toga, ne izračunava se optimalna brzina već se izračunava lokalna optimalna brzina, koja je uvek bezbedna za sledeći segment puta. Kao maksimalna brzina na kraju segmenta uzima se takva brzina pri kojoj je sigurno moguće potpuno ukočiti u toku sledećeg segmenta puta. Ovu brzinu je moguće izračunati pri ulasku u svaki segment puta za sledeći segment. Brzina skretanja se onda izračunava sa bezbednom brzinom kao gornjom granicom. Ovim se obezbeđuje da se u segment puta nikad ne uđe brzinom koja je prevelika u odnosu na sledeće segmente puta.



Slika 3.6: Primeri skretanja različitim ulaznim brzinama i izbora bezbedne brzine

Na osnovu pravaca agenta i pravca i dužine segmenta puta, određuje se maksimalna brzina na početku segmenta kojom je moguće dostići ključnu tačku na kraju segmenta. Bezbedna brzina se određuje izračunavanjem zaustavnih

puteva za različite brzine.

Zaustavni put se računa po formuli

$$s_z = v_s \cdot t$$

pri čemu je v srednja brzina u toku kočenja a t ukupno vreme kočenja. Agent ravnomerno usporava negativnim ubrzanjem a u toku celog kočenja, tako da su srednja brzina i vreme

$$v_s = \frac{v}{2}$$
$$t = \frac{v}{a}$$

pa je zaustavni put

$$s_z = \frac{v}{2} \cdot \frac{v}{a} = \frac{v^2}{2a}$$

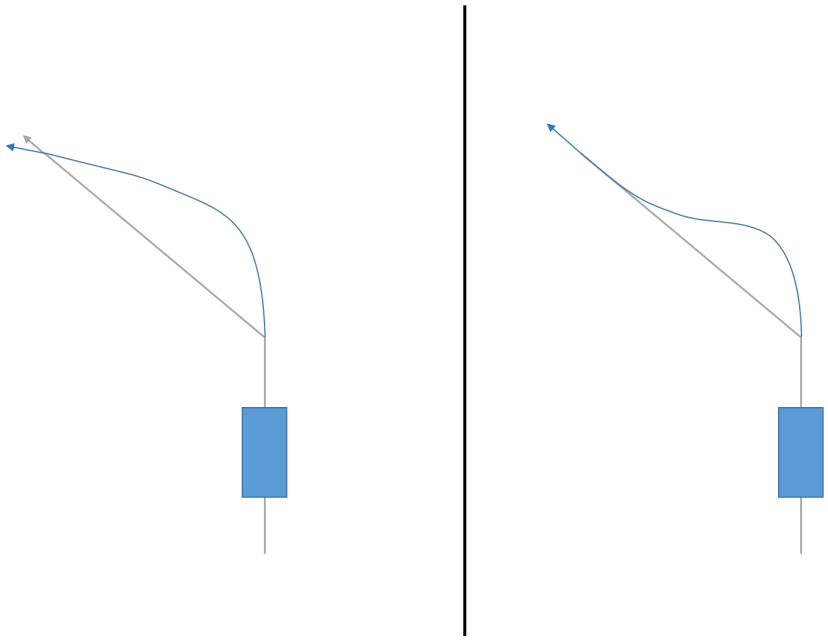
gde je v trenutna brzina agenta.

Najveća brzina pri kojoj je zaustavni put manji od dužine segmenta uzima se za bezbednu brzinu. Ovo izračunavanje se vrši samo jednom, po ulasku u segment puta, za sledeći segment puta. U svakom vremenskom intervalu u toku skretanja, koristeći bezbednu brzinu, proverava se da li je potrebno prilagoditi brzinu do kraja trenutnog segmenta puta. To se radi izračunavanjem kretanja do kraja trenutnog segmenta puta, uključujući i kočenje na kraju. Ukoliko je završna brzina manja od potrebne brzine, agent još uvek ne treba da počne sa kočenjem i željena brzina se ne menja. Ukoliko je završna brzina približna potrebnoj brzini, agent počinje da koči.

Ulazna brzina u segment puta je uvek takva da je moguće potpuno se zaustaviti do kraja segmenta (a često i znatno ranije). Pošto se provera kočenja izvršava u svakom vremenskom intervalu, nemoguće je premašiti krajnji momenat za bezbedno kočenje. Zbog toga će ovaj metod provere uvek odrediti dobar momenat za početak kočenja.

Izračunavanja vezana za praćenje puta i skretanje, uključujući izračunavanje bezbedne ulazne brzine u segment puta, pretpostavljaju da se, pri ulasku u segment puta, agent nalazi približno na koordinatama ključne tačke i usmeren u pravcu približnom pravcu prethodnog segmenta puta. Naivnim orijentisanjem prema sledećoj tački u toku kretanja krajnji pravac agenta odstupa od pravca korišćenog za izračunavanje bezbedne brzine. (slika 3.7 levo)

Da bi se osigurao očekivani pravac agenta na kraju segmenta puta, primenjuje se naglo skretanje. Umesto da se orijentiše prema sledećoj ključnoj tački na putu, agent će skrenuti više, kako bi se što pre vratio na pravac puta, a zatim će ispraviti pravac i pratiti put. Ugao naglog skretanja je u svakom trenutku dvostruko veći od trenutnog ugla između pravca agenta i puta, na taj način taj ugao se stalno smanjuje i dovodi do glatkog prilaska pravcu puta. (slika 3.7 desno)



Slika 3.7: Razlika između naivnog skretanja (levo) i naglog skretanja (desno)

3.4.3 Ponašanje prema ostalim učesnicima u saobraćaju

Drugi učesnici u saobraćaju ne utiču na pravac kretanja agenta, tako da je glavni problem u reagovanju na njih određivanje trenutne brzine. Na nižem nivou, ključno pitanje je "Da li u ovom momentu, uzevši u obzir ostale učesnike u saobraćaju u okolini, agent treba da počne da koči ili ne?"

Ukoliko je odgovor "da", agent će u datom vremenskom intervalu početi da koči. Ukoliko je odgovor "ne", agent u datom vremenskom intervalu neće reagovati na ostale učesnike u saobraćaju, već će se ponašati u skladu sa praćenjem puta. To, naravno, može uključivati kočenje iz drugih razloga ili ubrzavanje, ali to neće biti uzrokovano ostalim učesnicima u saobraćaju.

Agent neće početi da koči ranije nego što je to potrebno. U slučajevima kada je kočenje potrebna reakcija, agent će najpre proceniti zaustavni put pri trenutnoj brzini (videti poglavlje 3.4.2).

Ukoliko je zaustavni put znatno veći od daljine prepreke, agent neće početi da koči. Ukoliko je daljina prepreke bliska zaustavnom putu, agent će početi da koči. Radi dodatne bezbednosti, sva izračunavanja se vrše uzevši u obzir karakteristike agenta jedan ili dva vremenska intervala u budućnosti, dok se akcije korigovanja brzine izvršavaju u istom vremenskom intervalu. Na ovaj način se osigurava da će agent uvek moći pravovremeno da reaguje.

Izračunavanja su nezavisna u svakom trenutku. Brzina izračunata u odnosu na nekog učesnika u saobraćaju je željena brzina agenta i agent će početi da

ubrzava ili usporava kako bi joj se približio. Nakon akcije, u sledećem vremenskom intervalu, agent će ponovo uzeti u obzir novo stanje i krenuti sa procenama iz početka. Rezultati prethodnih izračunavanja ne utiču na trenutna. To znači da bi, na primer, korisnik aplikacije mogao da preuzme kontrolu od agenta, i da je preda nazad agentu kasnije. Agent će moći da nastavi samostalno kretanje istog momenta bez dodatne pripreme. Mogućnost preuzimanja kontrole od agenta nije implementirana u aplikaciji, ali ovakav dizajn znači da bi to bilo lako uraditi.

Agent ne uzima u obzir raspored puteva pri analiziranju kretanja drugih automobila, već samo njihove vidljive karakteristike. Na taj način se ne prave nikakve pretpostavke o budućem kretanju drugih automobila osim onoga što se već vidi, što daje veću fleksibilnost sistemu.

Vidno polje

Agenti imaju ograničeno vidno polje, daljinu van koje ne vide druge učesnike u saobraćaju. Sva pravila za interakciju sa drugim učesnicima u saobraćaju važe samo ako se isti nalaze unutar vidnog polja agenta.

Domet vidnog polja utiče na ponašanje agenta. Maksimalna brzina koju će agent dostizati će biti takva da zaustavni put agenta bude manji od dometa vidnog polja, kako bi bio u mogućnosti da reaguje na svaku promenu na rubu vidnog polja. Domet vidnog polja utiče i na interakciju sa pešačkim prelazima (videti poglavlje 3.4.3).

Semafori

Agent reaguje na semafore koji su ispred njega, okrenuti ka njemu i koji su pored puta kojim se agent trenutno kreće. Svi drugi semafori biće ignorisani. U zavisnosti od stanja semafora ponaša se na sledeći način:

- Zeleno svetlo - ignorisanje semafora
- Crveno svetlo - stajanje ispred semafora
- Žuto svetlo - ignorisanje ili stajanje u zavisnosti od situacije

Za reakciju na žuto svetlo, agent će proceniti svoju brzinu i daljinu od semafora i onda se odlučiti za stajanje ili ignorisanje semafora. Ukoliko agent može bezbedno da se zaustavi, zaustaviće se kada za to dođe vreme. U slučaju kada agent ne može da se zaustavi na vreme, na primer ukoliko je preblizu semaforu kada se uključi žuto svetlo, agent će ignorisati semafor i proći kroz raskrsnicu. Željena brzina agenta v_1 se računa kao

$$v_1 = \begin{cases} v_0 & \text{, ako je } d \geq s_z + d_{sem} \\ 0 & \text{, ako je } s_z < d < s_z + d_{sem} \\ v_0 & \text{, ako je } d \leq s_z \end{cases}$$

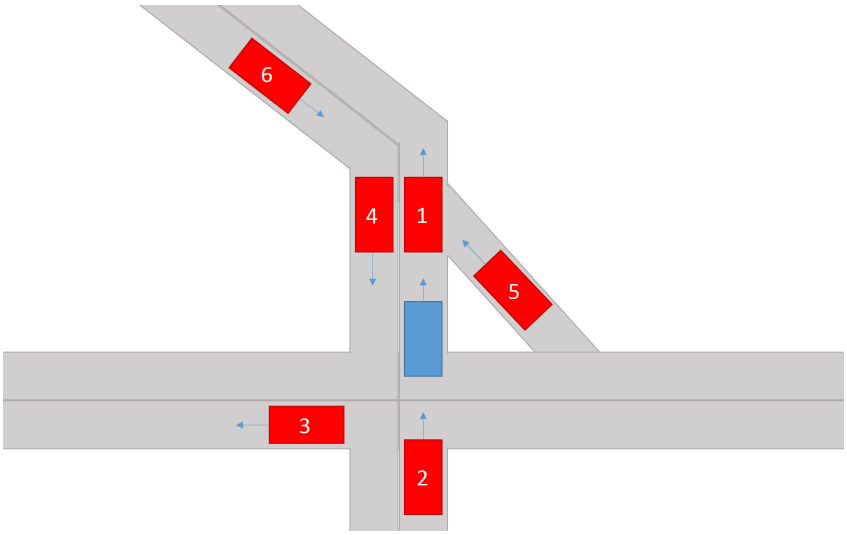
pri čemu je v_0 trenutna brzina agenta, d rastojanje agenta od semafora, d_{sem} rastojanje na kojem će agent stati ispred semafora, a s_z zaustavni put agenta

(videti poglavlje 3.4.2). Rastojanje d_{sem} postoji kako se ne bi desilo da agent prođe semafor u toku kočenja i ignoriše ga jer je sada semafor iza agenta.

Drugi automobili

Svi automobili u aplikaciji su autonomni agenti koji koriste iste algoritme za kretanje. Svaki automobil je nezavisan i ponaša se po istim pravilima. Svi automobili koriste iste algoritme za kretanje, praćenje puta i interakciju sa ostalim učesnicima u saobraćaju, ali ne koriste tu činjenicu da bi predvideli kretanje drugih automobila. Svaki automobil ima svoju kopiju mreže puteva i nema nikakve podatke o ostalim automobilima osim vidljivih karakteristika: pozicije, pravca kretanja i trenutne brzine.

Način reagovanja agenta na drugi automobil zavisi od prostornog odnosa agenta i automobila. Odluka koju agent donosi važi samo u tom vremenskom intervalu i isti automobil će ponovo biti razmatran u sledećem. Razlikujemo šest osnovnih slučajeva prostornih odnosa agenta i automobila: (slika 3.8)



Slika 3.8: Ilustracija 6 različitih slučajeva položaja automobila u odnosu na agenta

Automobil ispred agenta (na istom putu)

Ovaj slučaj se aktivira kada se automobil koji se razmatra nalazi ispred agenta i kreće se približno istim pravcem. Agent tada smatra da se automobil kreće istim putem kao i on. Reakcija agenta je potrebna kada je brzina agenta veća od brzine automobila. Željena brzina agenta, v_1, zavisi od brzine (v_2) i odstojanja automobila (d), odnosno od zaustavne daljine agenta s_z i daljine za koju agent može da izjednači brzinu automobila s_2:

$$s_z = \frac{v_1^2}{2a}$$
$$s_2 = \frac{\Delta v^2}{2a}$$

pri čemu je Δv razlika brzina agenta i automobila. Formula je zasnovana na istom principu kao formula za zaustavni put (videti poglavlje 3.4.2), s tim što se u ovom slučaju razmatra ubrzanje, odnosno usporenje, od v_1 do v_2, pa se kao brzina u formuli uzima razlika brzina agenta i automobila Δv. Željena brzina agenta se zatim računa kao

$$v_1 = \begin{cases} v_0 & \text{, ako je } d > s + s_2 + d_{min} \\ v_2 & \text{, ako je } s + s_2 + d_{min} < d < s + c \\ 0 & \text{, ako je } d < s_2 + c \end{cases}$$

pri čemu je v_0 trenutna brzina agenta a d_{min} minimalno dozvoljeno odstojanje dva automobila, konstanta zadata konfiguracionom datotekom.

Pošto se izračunavanje ponavlja u svakom vremenskom intervalu i uzima u obzir promene u brzini automobila, u praksi će agent, kada se približi automobilu, izjednačiti brzinu sa njim. Ukoliko automobil ubrza ili uspori, agent će ga pratiti, sve vreme održavajući bezbedno rastojanje.

Ovaj slučaj takođe pokriva i situaciju kada se automobil ne kreće istim pravcem kao agent, ali se nalazi direktno ispred njega (na primer u slučaju zastoja u raskrsnici). U toj situaciji agent će zakočiti i stajati dok se automobil ne pomeri.

Automobil direktno iza agenta

U ovom slučaju agent smatra da se automobil kreće istim putem kao i on, ali je iza agenta. Automobil ne utiče na kretanje agenta, pa ga agent ignoriše.

Automobil se udaljava od agenta

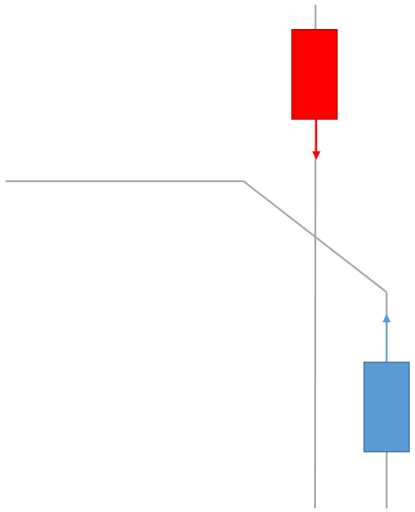
Kada se automobil udaljava od agenta u datom vremenskom intervalu, kretanje automobila ne utiče na kretanje agenta. U ovom slučaju agent ignoriše automobil. Ignorisanje automobila je bezbedna odluka čak i u slučaju promena kretanja automobila, jer će isti automobil ponovo ući u razmatranje u sledećem vremenskom intervalu. Ako se situacija promeni, agent će reagovati drugačije.

Automobil koji se kreće paralelno sa agentom (u suprotnom smeru)

U ovom slučaju agent ignoriše automobil. Pretpostavlja se da se automobil kreće istim putem kao agent ali u suprotnom smeru. Kako prestrojavanje (promena trake na putu) nije pokriveno modelom, mimoilaženje sa automobilom je uvek bezbedno. Posebna pažnja se

obraća, osim na pravac i ugao kretanja, i na rastojanje pravaca kretanja automobila i agenta, kako bi se ovaj slučaj razdvojio od opšteg slučaja.

Važan podslučaj ovog slučaja je situacija u kojoj će agent promeniti pravac kretanja pre mimoilaženja sa automobilom. Skretanje levo u raskrsnici je konkretan primer (slika 3.9).



Slika 3.9: Primer specijalnog slučaja skretanja levo u raskrsnici

Ova situacija se prepoznaje kada je udaljenost automobila veća od udaljenosti agenta do sledeće ključne tačke na putu. Razmatra se presek sledećeg segmenta puta (pravca agenta posle skretanja) i pravca kretanja automobila. Ukoliko taj presek postoji i automobil ga još uvek nije prošao, agent će ukočiti pre promene pravca i stajati dok ga automobil ne mimoiđe. Izuzetak se pravi ako automobil stoji, na primer na semaforu u raskrsnici kroz koju agent skreće. U tom slučaju agent neće čekati automobil.

Opšti slučaj preseka pravaca agenta i automobila

Automobil koji nije klasifikovan pod neki od prethodnih slučajeva spada pod opšti slučaj.

U opštem slučaju važi "pravilo desne strane": automobilu koji je sa desne strane treba ustupiti prvenstvno prolaza, automobilu koji je sa leve strane ne treba ustupiti prvenstvo prolaza. Razmatra se presek pravaca kretanja agenta i automobila. Za određivanje da li je automobil levo ili desno dovoljno je proveriti da li je ugao agent - presek - automobil veći ili manji od 180°.

Ukoliko je automobil sa leve strane, agent ima prvenstvo prolaza i ignoriše automobil. Ukoliko je automobil sa desne strane, agent će stati i

propustiti automobil, osim ako je automobil dovoljno daleko da mu agent neće biti smetnja. U tu svrhu agent će proceniti daljinu automobila od preseka u momentu kada agent stigne do njega (s_u), pod pretpostavkom da automobil stalno ubrzava (najgori slučaj): ova formula je slična formuli za zaustavni put (videti poglavlje 3.4.2) : daljina ubrzanog kretanja je jednaka daljini ravnomernim kretanjem srednjom brzinom za isto vreme.

$$d_{a2} = d_{a1} - s_u$$
$$s_u = v_{s2} \cdot t$$
$$v_{s2} = \frac{v_a + a_a \cdot t}{2}$$

pri čemu je d_a daljina automobila od preseka, v_a i a_a trenutna brzina i ubrzanje automobila i t vreme za koje će agent stići do preseka. Vreme t se računa kao $t = \frac{d}{v}$, gde je d daljina agenta do preseka a v trenutna brzina agenta.

Procena uzima u obzir najgori slučaj, a to je da automobil konstantno ubrzava dok se agent kreće konstantnom brzinom. Ako pri ovakvoj proceni automobil bude dovoljno blizu ispred preseka u vreme kada agent stigne do njega, agent će početi da koči kako bi propustio automobil. Ako agent proceni da će automobil proći presek ili još uvek biti dovoljno daleko od njega da će agent proći presek pre susreta, agent će ignorisati automobil.

Automobil van trenutnog segmenta puta

Ovo nije zaseban slučaj sam za sebe, ali menja način izračunavanja za ostale slučajeve.

Ovaj slučaj se aktivira ukoliko se automobil nalazi dalje od sledeće ključne tačke na putu, odnosno od početka sledećeg segmenta puta. Pošto se prelaskom na drugi segment puta može promeniti pravac kretanja agenta, a način reagovanja agenta na automobil zavisi od odnosa pravca kretanja agenta i automobila, za određivanje odnosa automobila i agenta će biti korišćeni očekivani pravac i brzina agenta na početku sledećeg segmenta puta. U toku trenutnog segmenta puta agent će održavati brzinu takvu da može da ukoči do kraja trenutnog segmenta puta kako bi mogao da reaguje na eventualne promene u narednim vremenskim trenucima.

Pešaci i pešački prelazi

Kao i sa automobilima, agent ne zna u kom se stanju nalazi pešak, niti koja mu je destinacija. Od podataka vezanih za pešaka koristi samo njegovu poziciju, brzinu i pravac kretanja.

U opštem slučaju, agent će pešacima koji prelaze put ustupiti prvenstvo prolaska. To će učiniti tako što će ukočiti ispred mesta gde pravac kretanja pešaka seče pravac kretanja agenta. Način ponašanja je sličan kao opšti slučaj

reagovanja na automobile, s tim što će agent ustupiti prvenstvo prolaza pešacima i sa leve i sa desne strane, kao i pešacima koji stoje pored puta. Pešaci koji se nalaze iza agenta i pešaci koji se udaljavaju od agenta se ignorišu.

Agent računa presek pravca kretanja pešaka i svog pravca kretanja, a zatim procenjuje daljinu pešaka u vreme kada agent stigne do preseka:

$$d_{p2} = d_{p1} - \frac{d}{v_0} \cdot v_p$$

gde je d daljina agenta do preseka, d_{p1} daljina pešaka do prelaza i v_p brzina pešaka. Agent će zatim odrediti da li je potrebno kočiti kako bi propustio pešaka.

$$v_1 = \begin{cases} v_0 & \text{, ako je } |d_{p2}| > d_{min} \\ 0 & \text{, ako je } |d_{p2}| \leq d_{min} \end{cases}$$

gde je d_{min} minimalno bezbedno rastojanje pešaka i automobila, konstanta zadata konfiguracionom datotekom. Ako će pešak biti dovoljno daleko od agenta kada on stigne do preseka, agent će ignorisati pešaka. U suprotnom, agent će ukočiti ispred preseka.

Ukoliko se pešak kreće prema pešačkom prelazu koji se nalazi ispred agenta, agent će ukočiti ispred tog prelaza. Izuzeci od ovog ponašanja su u situaciji kada je pešački prelaz zatvoren semaforom ili ako je pešak dovoljno daleko od puta da će agent proći mesto prelaska pre nego što pešak stigne do njega. Za tu procenu koristi se trenutna brzina pešaka v_p

$$v_1 = \begin{cases} v_0 & \text{, ako je } \frac{d}{v_0} \cdot v_p + d_{min} < d_{pp} \\ 0 & \text{, inače} \end{cases}$$

gde je d daljina prelaza, d_p i v_p daljina i brzina pešaka, d_{pp} daljina pešaka od prelaza a d_{min} minimalno bezbedno rastojanje pešaka i automobila.

Specijalni slučajevi pokrivaju situacije kada je agent previše blizu tački susreta sa pešakom i time blokira pešaka da pređe put. U takvoj situaciji agent će prestati da čeka pešaka.

Kao i kod reagovanja na automobile, i za pešake se izračunavanje menja ukoliko se tačka susreta nalazi van trenutnog segmenta puta agenta. Tada se predviđa pozicija i pravac pešaka i agenta u vreme prelaska na sledeći segment puta i reaguje se na osnovu toga.

Kada se pešački prelaz nađe u dometu agenta, čak i bez vidljivih pešaka u okolini, agent će početi da prilagođava brzinu tako da može da se zaustavi ispred prelaza. Agent će nastaviti da se kreće smanjenom brzinom sve dok ne bude video dovoljnu okolinu prelaza da osigura da na prelazu neće biti pešaka. Poluprečnik bezbedne okoline je

$$r = v_p \cdot \frac{d}{v_0}$$

pri čemu je v_p maksimalna brzina pešaka a d daljina od agenta do prelaza. U periodu dok okolina prelaza nije potpuno vidljiva agentu, on će održavati

bezbednu brzinu, takvu da je zaustavni put agenta kraći od trenutne daljine prelaza.

$$v_1 = \begin{cases} v_0 & \text{, ako je } d > s_z + d_{min} \\ 0 & \text{, ako je } d < s_z + d_{min} \end{cases},$$

pri čemu je d_{min} minimalno odstojanje između automobila i pešaka, konstanta zadata konfiguracionom datotekom, a s_z zaustavni put agenta (videti poglavlje 3.4.2). Na taj način, ukoliko se u okolini prelaza pojavi pešak koji se kreće prema prelazu, agent će biti u stanju da se zaustavi i propusti pešaka. Kada cela okolina prelaza bude vidljiva agentu i u njoj nema pešaka koji se kreću ka prelazu, agent će ignorisati prelaz.

Ako se pešački prelaz nalazi pored semafora, da li će agent reagovati na njega ili ne zavisi od stanja semafora. Prelaze koji su za pešake zatvoreni semaforom agent će ignorisati.

3.4.4 Ažuriranje parametara (pozicije, pravca i brzine)

Kretanje agenta uključuje određivanje pozicije i pravca agenta u diskretnim vremenskim razmacima. Oni se određuju na osnovu karaketeristika kao što su trenutna brzina i pravac.

Za određivanje brzine u svakom trenutku ("željena brzina") postoji nekoliko nivoa ograničenja. Podrazumevano, agent želi da se kreće najvećom mogućom brzinom. Ukoliko nema drugih ograničenja, agent će ubrzati do te brzine.

Prvo ograničenje je vidno polje agenta. Iako je maksimalna brzina zadata karakteristika agenta, brzina koju će agent pokušati da postigne će biti ograničena tako da potreban zaustavni put uvek bude unutar vidnog polja, odnosno tako da je

$$s_z < r_s$$

gde je s_z zaustavni put agenta (videti poglavlje 3.4.2) a r_s poluprečnik vidnog polja agenta. Na taj način se osigurava da agent može da reaguje na svaku prepreku koju primeti.

Sledeće ograničenje je ugao između pravca kojim se agent trenutno kreće i pravca kojim treba da se kreće. Agent će prilagoditi željenu brzinu tako da može da dostigne željeni pravac pre kraja trenutnog segmenta puta, odnosno tako da je

$$\frac{\delta}{\delta_{max}} \geq \frac{d}{v \cdot \Delta t}$$

gde je δ razlika pravaca a δ_{max} maksimalni ugao skretanja u jednom vremenskom intervalu. Ovo je samo gruba procena gornje granice brzine, koja razmatra isključivo karakteristike agenta: brzinu, daljinu i ugao skretanja u jedinici vremena. Ova procena služi samo da ubrza sledeće korake izračunavanja tako što smanjuje prostor pretrage ograničavanjem maksimalne brzine.

U obzir se uzima i ulazna brzina za sledeći segment puta ("bezbedna brzina"). Izračunavanje bezbedne brzine je objašnjeno u poglavlju 3.4.2. Za željenu brzinu agenta uzima se najmanje od svih ograničenja. Tu brzinu će agent pokušati da postigne ukoliko ne bude morao da ukoči zbog nekog od drugih učesnika u saobraćaju.

Na osnovu trenutnih parametara agenta i izračunate željene brzine i pravca određuje se ubrzanje i ugao skretanja agenta za sledeći vremenski interval:

Ubrzanje, odnosno promena brzine do sledećeg vremenskog intervala, se računa kao

$$a = \begin{cases} max(a_{max}, \frac{v - v_0}{\Delta t}) & \text{, ako je } v > v_0 \\ -max(a_k, \frac{v_0 - v}{\Delta t}) & \text{, ako je } v < v_0 \\ 0 & \text{, ako je } v = v_0 \end{cases}$$

gde je a_{max} maksimalno ubrzanje agenta, a_k maksimalna sila kočenja, a v_0 željena brzina agenta. Δt je jedan vremenski interval.

Ugao skretanja se računa kao

$$\alpha = \begin{cases} max(\alpha_{max}, \varphi_z - \varphi) & \text{, ako je } \varphi < \varphi_z < \varphi + \pi \\ -max(\alpha_{max}, \varphi - \varphi_z) & \text{, ako je } \varphi > \varphi_z > \varphi - \pi \end{cases}$$

gde je φ trenutni pravac agenta, φ_z željeni pravac, a α_{max} maksimalni ugao skretanja u jednom vremenskom intervalu.

Na osnovu izračunatog ubrzanja i ugla skretanja se računaju nova brzina, pravac i pozicija agenta. Pravac agenta se svodi na ugao između 0 i 360°, odnosno 0 i 2π radijana.

$$v = v + a \cdot \Delta t$$
$$\varphi = \varphi + \alpha$$
$$(x, y) = (x + \cos\varphi \cdot v, y + \sin\varphi \cdot v)$$

Kretanje se diskretizuje tako što se najpre vrši promena pravca, a zatim se dodaje brzina u novom pravcu. Za mali razmak između vremenskih intervala ovo se pokazalo dovoljno dobrom aproksimacijom kretanja.

3.5 Dijagnostika

U toku rada aplikacija snima podatke o kretanju agenta, kao i drugih učesnika u saobraćaju. Ovi podaci se mogu koristiti za analizu ponašanja agenta u slučaju da dođe do neke nepredviđene situacije. U toku rada aplikacije ovi podaci se mogu iscrtati na ekranu ili ispisati u komandnoj liniji.

Agent čuva istoriju kretanja, što uključuje pozicije, pravce i trenutne brzine kretanja. Isto važi i za druge automobile i pešake, pri čemu pešaci čuvaju i istoriju stanja konačnog automata. Pored toga, agent beleži istoriju interakcija sa svakim automobilom i pešakom kojeg sretne. Ove informacije, zajedno sa informacijama koje su zabeležili ostali učesnici u saobraćaju, daju potpunu sliku

ponašanja agenta i u krajnjem slučaju omogućavaju potpunu rekonstrukciju situacije.

Svi podaci se snimaju kao nizovi sa elementima za svaki vremenski interval, za proizvoljan broj vremenskih intervala. U trenutnoj implementaciji čuva se između 500 i 600 elemenata, što je oko 10 sekundi istorije.

3.6 Performanse sistema

Hardverska zahtevnost aplikacije zavisi od broja automobila i pešaka, pri čemu su izračunavanja koja vrše automobili zahtevnija od izračunavanja koja vrše pešaci. U poređenju sa njima, izračunavanja vezana za ostatak aplikacije (uključujući iscrtavanje) su zanemarljiva.

Frekvencija osvežavanja aplikacije je fiksirana na 50 puta u sekundi, tako da je metrika koja je korišćena maksimalni broj automobila i pešaka pri kojem aplikacija održava tu frekvenciju bez usporenja. Taj broj je različit u zavisnosti od hardvera na kojem se aplikacija pokreće, konkretno radnog takta procesora i memorije. Sledi nekoliko primera različitih računara i kombinacija broja automobila i pešaka na njima.

Konfiguracija 1: računar sa procesorom Intel® Core™ 2 Duo E7200 (radni takt 2,53 GHz), radni takt memorije 400 MHz

Agenti	Pešaci
1	850
15	540
30	375
45	285
60	195
75	145
90	90
105	65
120	30
135	0

Konfiguracija 2: laptop sa procesorom Intel® Core™ i5-3210M (radni takt 2,50 GHz), radni takt memorije 800 MHz

Agenti	Pešaci
1	700
20	420
40	280
60	200
80	140
100	85
120	40
140	0

Konfiguracija 3: računar sa procesorom Intel® Core™ i5-4440 (radni takt 3,10 GHz), radni takt memorije 1600 MHz

Agenti	Pešaci
1	1000
25	575
50	350
75	250
100	150
125	100
150	50
175	0

Memorijski zahtevi aplikacije takođe zavise od broja učesnika u saobraćaju. Dodatna količina memorije se uglavnom koristi za putanje agenata i tabele istorije svakog agenta sa svakim automobilom i pešakom. Slede primeri ukupnih memorijskih zahteva za nekoliko kombinacija broja automobila i pešaka. Merenje je izvršeno na računaru sa 6 GB radne memorije, sa operativnim sistemom Microsoft® Windows® 8.1.

Agenti	Pešaci	Memorija
1	0	15 - 17 MB
1	100	19 - 19 MB
1	1000	35 - 35 MB
50	0	22 - 23 MB
50	100	23 - 24 MB
50	1000	40 - 42 MB
100	0	27 - 30 MB
100	100	31 - 50 MB
100	100	30 - 48 MB
100	1000	47 - 69+ MB

3.7 Pregled C++ klasa i metoda

Aplikacija je objektno-orijentisana, sa objektima koji odgovaraju svim elementima modela. Na vrhu hijerarhije je `Svet`, koja sadrži i koordinira svim elementima aplikacije. Klasa `Svet` takođe sadrži i glavnu petlju aplikacije, koja ažurira i iscrtava sve aktivne objekte.

Elementi modela su objekti odgovarajućih klasa: `Agent`, `Pesak`, `Semafor`.

Klasa `Agent` sadrži sve metode za praćenje puta i interakciju sa ostalim učesnicima u saobraćaju. Izračunavanje putanje agenta se vrši odvojeno, u mreži puteva. `Agent` sadrži mrežu puteva i od nje dobija trenutni put u putanji. `Agent` je takođe zadužen i za čuvanje dijagnostičkih informacija, svoje istorije kretanja i rezultata prethodnih interakcija sa drugim automobilima i pešacima.

Klasa `Pesak` sadrži instancu konačnog automata koji kontroliše kretanje pešaka. Svaki `Pesak` sadrži stanje u kojem se nalazi. Kretanje u zavisnosti

od stanja, kao i promene stanja u zavisnosti od situacije u kojoj se pešak nalazi, vrše se interno u klasi. `Pesak` u sebi takođe sadrži i skup izdvojenih puteva koji koristi za efikasnije izračunavanje (videti poglavlje 3.3). Kao i agenti, i pešaci čuvaju istoriju svog kretanja.

Klasa `Semafor` se smatra aktivnim objektom jer ažurira svoje stanje. `Semafor` sadrži samo svoje trenutno stanje i metod za promenu stanja. Promena stanja semafora je sinhronizovana između svih semafora tajmerom koji dolazi od klase `Svet`.

Mreža puteva je implementirana kroz klase `Mreza`, `Raskrsnica`, `Put`, `Prelaz` i `Tacka`.

Klasa `Mreza` u sebi sadrži raskrsnice i puteve. Osim toga, mreža je zadužena za izračunavanje putanje agenta primenom algoritma A*. Mreža u sebi formira niz puteva koji čine putanju i predaje ih agentu po potrebi. Svaki `Agent` ima svoju instancu mreže, ali postoji jedna glavna `Mreza` koja je zadužena za dodavanje agenata i pešaka. Pešaci se dodaju pri pokretanju aplikacije na nasumična mesta van puteva, dok se novi agenti dodaju na slobodne raskrsnice. Dodavanje agenata može trajati duže u zavisnosti od broja slobodnih raskrsnica.

Klase `Raskrsnica` i `Put` su kontejner klase, klase koje sadrže puteve, odnosno ključne tačke. Raskrsnice pored puteva koji je povezuju sa drugim raskrsnicama sadrži i unutrašnje puteve, semafore i pešačke prelaze koji joj pripadaju.

4. Zaključci i dalji rad

Cilj ovog rada je bio da se što jednostavnijim algoritmima pokrije jednostavan model. To je urađeno uspešno, ali se pokazalo da ovaj pristup nije dovoljno efikasan. Na prvi pogled dovoljni, jednostavni algoritmi su se vrlo brzo zakomplikovali specijalnim slučajevima i podslučajevima, tako da je kod postao veoma težak za razumevanje i održavanje.

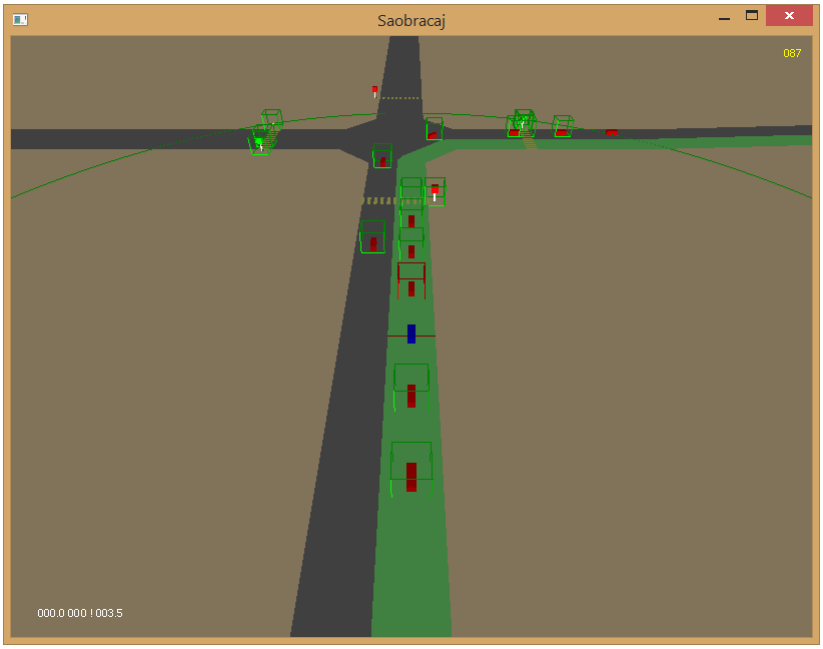
Rad na ovoj aplikaciji doveo je do uočavanja mnogih detalja funkcionisanja saobraćaja koji na prvi pogled nisu bili očigledni. To se naročito uočava u interakciji agenta sa drugim učesnicima u saobraćaju. Nepredviđene situacije na koje se naišlo u toku razvoja i njihova rešenja u aplikaciji, potvrdili su složenost sistema koji se modeluje. Konačni model saobraćaja i implementacija aplikacije, iako drastično pojednostavljeni, ipak stvaraju svest o tome koliko je kompleksan problem razvoja sistema za automatsku vožnju.

Najveći izazov u toku razvoja aplikacije je bilo reagovanje agenta na druge automobile. Određivanje različitih nezavisnih slučajeva koji pokrivaju sve moguće situacije je zahtevalo veliku količinu analiziranja i testiranja. Pošto su automobili u stalnom pokretu, odnos između njih i agenta se stalno menja. Zato je bilo potrebno obratiti posebnu pažnju na granične slučajeve i neočigledne specijalne situacije. Svi slučajevi su međusobno povezani, tako da su fina podešavanja jednog slučaja često zahtevala izmene i na ostalima.

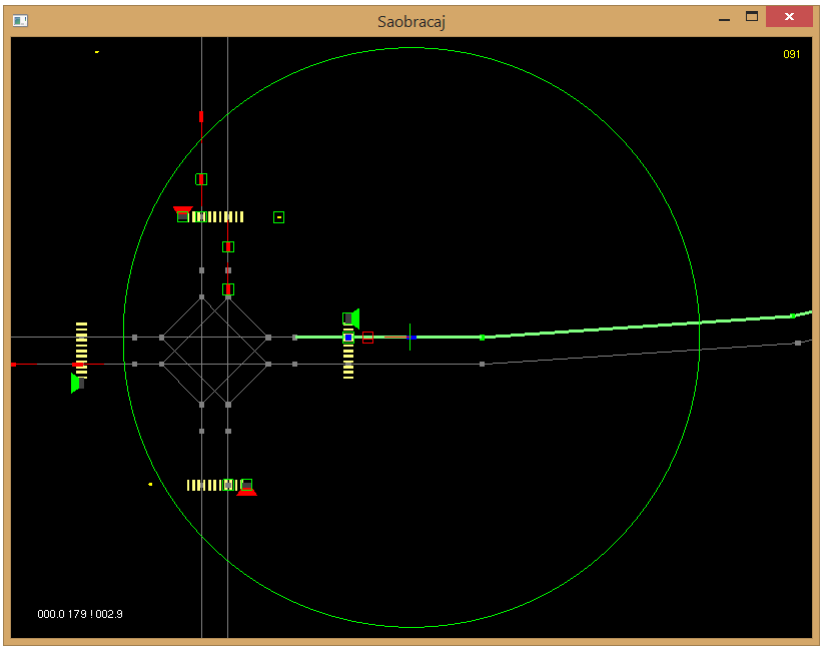
Jedan od najvećih izazova u toku rada je predstavljalo sistematsko testiranje. Dijagnostički alati su proširivani po potrebi i time su uvek kasnili za potrebama aplikacije. Testiranje je uglavnom bilo ručno, što je usporavalo rekonstrukciju i rešavanje problema. Nepreglednost koda izazvana mnogim zakrpama algoritama je dodatno usporila ispravljanje grešaka, na šta je istrošen najveći deo vremena u toku rada.

Aspekt koji nije pokriven u aplikaciji je mogućnost paralelizacije. Ne samo da su automobili i pešaci nezavisni jedni od drugih, već je i većina izračunavanja od strane jednog agenta nezavisna, i kao takva veoma dobra za paralelizaciju. Jedina veza između reagovanja agenta na različite učesnike u saobraćaju u trenutnoj implementaciji je korišćenje prethodno izračunate brzine za preskakanje nekih izračunavanja, što u paralelnoj implementaciji ne bi bilo potrebno. Uz odbacivanje te zavisnosti, reagovanja na učesnike u saobraćaju kao i izračunavanja vezana za praćenje puta bi mogla da se izvršavaju potpuno paralelno.

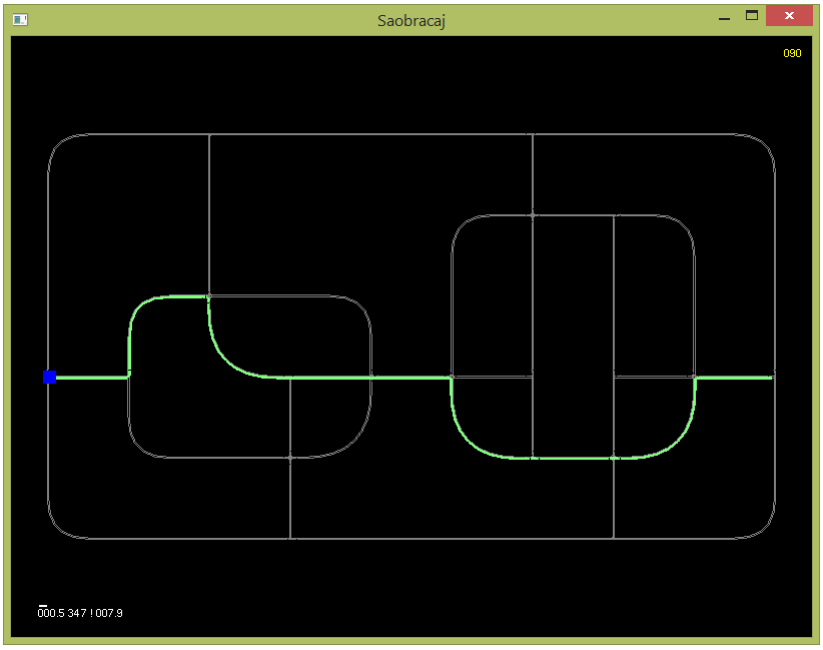
Dodatak A. Snimci ekrana aplikacije



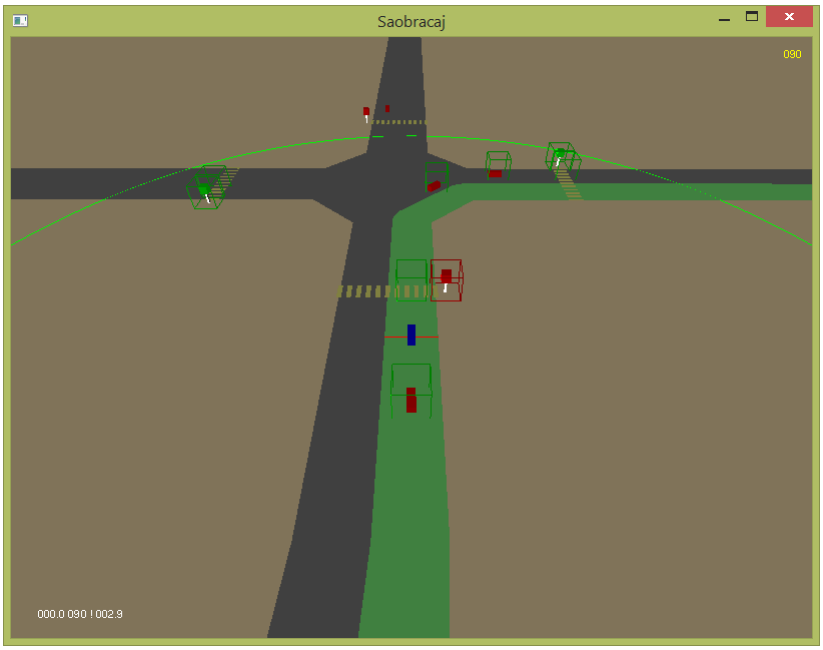
Slika A.1: Izgled aplikacije sa tačke gledišta agenta



Slika A.2: 2D prikaz aplikacije

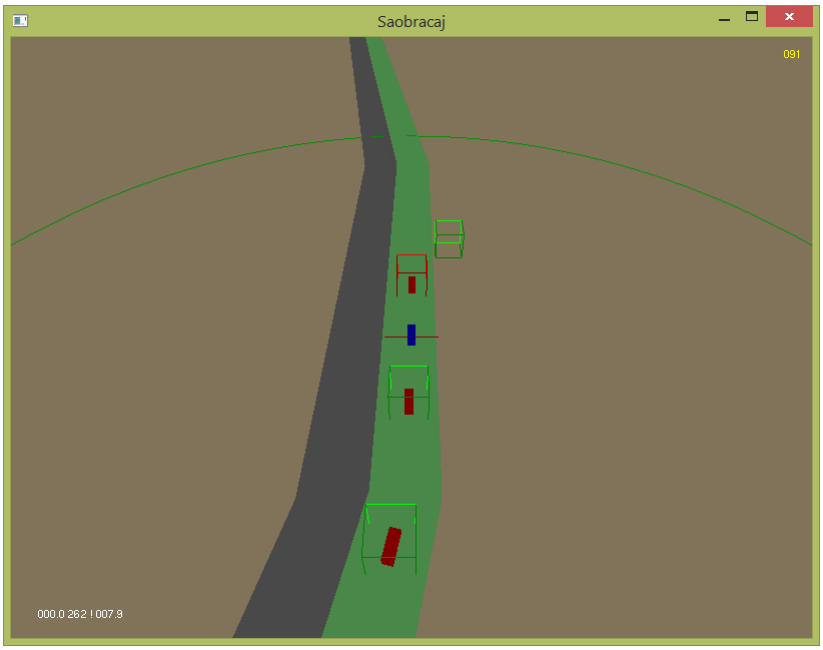


Slika A.3: Primer izabrane putanje na mreži puteva



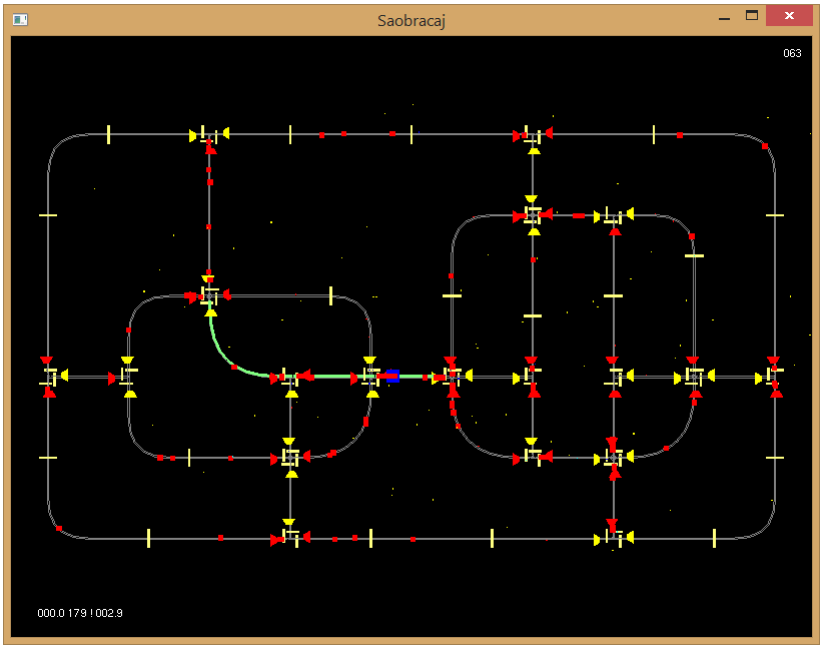
Slika A.4: Agent stoji zbog crvenog svetla na raskrsnici

Slika A.5: Agent propušta pešaka na pešačkom prelazu



Slika A.6: Agent stoji iza automobila koji propušta pešaka

Slika A.7: Agent čeka automobil iz suprotnog smera pre nego što skrene levo u raskrsnici



Slika A.8: Dijagnostički prikaz aplikacije koji prikazuje sve učesnike u saobraćaju

Bibliografija

[1] Stanje bezbednosti saobraćaja u Republici Srbiji,
http://prezentacije.mup.gov.rs/usp/Statistika/Statistika.html

[2] "An automated adventure at the wheel of a driverless BMW",
http://www.thenational.ae/business/technology/
an-automated-adventure-at-the-wheel-of-a-driverless-bmw
The National Business,
septembar 2014

[3] "Who's Behind the Wheel? Nobody.",
http://www.wsj.com/articles/SB10000872396390443524904577651552635911824
The Wall Street Journal,
septembar 2014

[4] "Ford is ready for the autonomous car. Are drivers?",
https://gigaom.com/2012/04/09/ford-is-ready-for-the-autonomous-car-are-drivers/
GigaOM Media,
april 2012

[5] "Toyota sneak previews self-drive car ahead of tech show",
http://www.bbc.com/news/technology-20910769
BBC,
januar 2014

[6] "Elon Musk Says Self-Driving Tesla Cars Will Be in the U.S. by Summer"
http://www.nytimes.com/2015/03/20/business/
elon-musk-says-self-driving-tesla-cars-will-be-in-the-us-by-summer.
html
The New York Times,
mart 2015

[7] Darpa Urban Challenge 2007,
http://archive.darpa.mil/grandchallenge/

[8] The latest chapter for the self-driving car: mastering city street driving,
http://googleblog.blogspot.com/2014/04/
the-latest-chapter-for-self-driving-car.html

Zvanični Google blog,
april 2014.

[9] Predrag Janičić, Mladen Nikolić,
Veštačka inteligencija,
juni 2010.

[10] OpenGL,
https://www.opengl.org/

[11] The freeglut Project,
http://freeglut.sourceforge.net/