SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1704

PRIMJENA NEURONSKIH MREŽA ZA RAZVOJ GLOBALNE STRATEGIJE ROBOTSKOG NOGOMETA

Edin Mangić

Zagreb, rujan 2009.

Zahvaljujem prof. dr. sc. Ivanu Petroviću i dipl. ing. Mišelu Brezaku na stručnom vodstvu i podršci tijekom izrade diplomskog rada.

Zahvaljujem roditeljima na podršci tijekom cijelog studija.

Sadržaj

1. Uvod

Umjetna inteligencija je područje računarstva koje se paralelno s pojavom digitalnih računala intenzivno istražuje već četrdeset godina. Danas su inteligentni sustavi postali neizostavni dio tehnoloških industrijskih procesa, upravljačkih sustava i istraživačkih alata. Medicinska dijagnostika se sve više oslanja na zaključke ekspertnih sustava, a velika tranzitna čvorišta i skladišni sustavi gotovo i ne bi mogli funkcionirati bez optimizacije i planiranja utemeljenog na evolucijskim algoritmima.

Alan Turing[1] na pitanje što je to umjetna inteligencija odgovara tezom: «Ako se stroj ponaša inteligentno poput ljudskog bića, tada on i jest inteligentan poput ljudskog bića»[2], što bi značilo da u krajnosti stupanj inteligencije stroja možemo mjeriti samo kroz njegovo ponašanje i samostalnost.

Da bi inteligentni sustav bio samostalan mora biti u mogućnosti samostalno donositi odluke utemeljene na prethodno stečenom znanju, te također biti u stanju prepoznati, obraditi i pohraniti novo znanje, kako bi u budućnosti mogao donositi bolje odluke.

U ovome se radu istražuje mogućnost primjene neuronskih mreža na upravljanje višerobotskim sustavima. Cilj je izgraditi sustav koji će samostalno i postupno sakupljati znanje i na temelju njega donositi odluke te biti uspješniji u natjecanju robotskog nogometa.

[1] Engleski matematičar i istraživač, autor «Turingovog stroja»
[2] Eng. «*If a machine acts as intelligently as a human being, then it is as intelligent as a human being*», početna teza Turingovog testa.

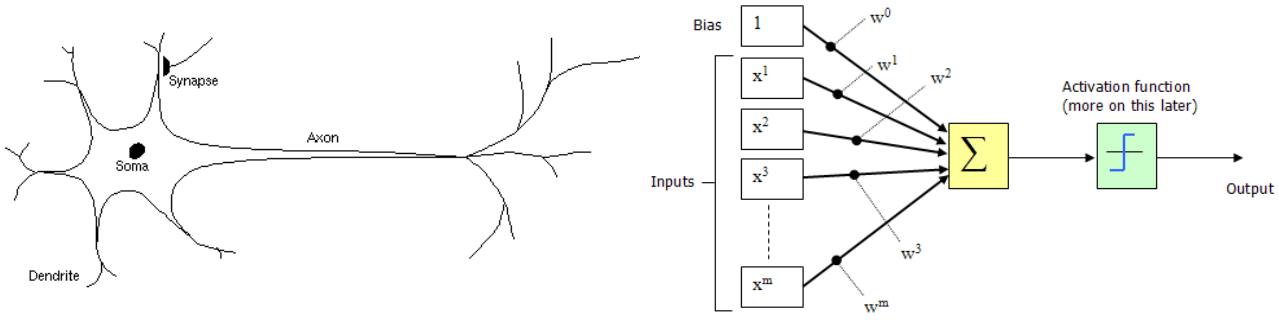
2. Neuronske mreže

2.1 Koncept neuronske mreže

Umjetne neuronske mreže[3] inspirirane su biološkim neuronskim mrežama životinjskih vrsta i medicinskim istraživanjima na ljudskom mozgu. Sami neuroni od kojih se sastoji neuronska mreža oponašaju biokemijske procese u biološkom neuronu, a propagiranjem signala kroz veze među neuronima oponašaju se sinapse i neuronska interakcija živčanog sustava.

Motivacija za izradu umjetne neuronske mreže leži u činjenici da je ljudski mozak, iako nekoliko puta sporiji od digitalne elektroničke logike[4], i dalje vrlo uspješan u rješavanju izrazito složenih problema kao što su prepoznavanje lica, razumijevanje semantike prirodnog govora te predviđanja i klasifikacije na temelju iskustva. Ljudski mozak ostvaruje također i visok stupanj paralelizma i nelinearnosti.

Zbog jednostavnog matematičkog modela neurona, neuronske mreže su pogodne za implementaciju u računalu.



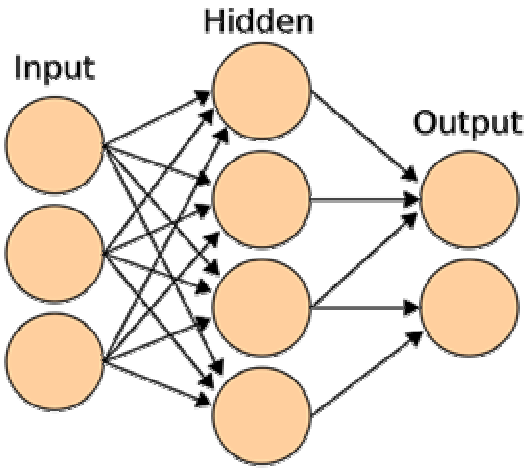
Slika 1. Biološki neuron se aproksimira matematičkim modelom

Slično biološkom neuronu, matematički model neurona prima signale od drugih neurona na svoj ulaz te ih sumirane predaje aktivacijskoj funkciji. Aktivacijska funkcija određuje kakav će biti izlaz tog neurona, te vrijednost izlaza prosljeđuje dalje u

[3] U nastavku teksta: neuronske mreže
[4] Ograničenu brzinu mozak nadoknađuje ogromnim brojem neurona. Ljudski mozak sadrži oko 10 milijardi neurona i oko 60000 milijardi neuronskih veza.

mrežu. Primljeni signal se oslabi ili ojača ovisno o snazi i stupnju povezanosti s prethodnim neuronom iz kojega dolazi signal, tako da će jača veza značiti i jaču aktivaciju jezgre neurona. Da bi se neuron aktivirao, ulazni signal mora prijeći odrađenu vrijednost praga. Svaki neuron djelomično obrađuje signal, a njihovo međudjelovanje realizira svojstva poput nelinearne aproksimacije i generalizacije.

Postoji nekoliko tipova neuronskih mreža, s različitim tipovima modela neurona i različitim topologijama mreže. U okviru ovog rada razmatrati će se tip mreže koji se zove višeslojni perceptron.



Slika 2. Višeslojni perceptron

Višeslojni se perceptron sastoji od tri sloja neurona: ulazni sloj, skriveni sloj i izlazni sloj neurona. Tijekom propagacije kroz neuronsku mrežu, ulazni vektor prolazi paralelnu obradu u slojevima mreže. Izlaz višeslojnog perceptrona je također vektor.

2.2. Svojstva neuronskih mreža

Neuronska mreža se s matematičkog stajališta može promatrati kao nelinearna aproksimacijska funkcija više varijabli.

Obrada signala u neuronu definira se formulom:

$$u = \sum wx$$

Izlaz svakog neurona definira se formulom:

$$y = \varphi(u - \Theta)$$

Ulazna funkcija koju neuronska mreža treba naučiti opisana je parovima ulaz-izlaz. Tijekom procesa treniranja neuroni u neuronskoj mreži mijenjaju vrijednosti težina i pragova, čime se postiže bolja aproksimacija i generalizacija ulazne funkcije. Kada je proces treniranja gotov, očekuje se da će neuronska mreža i u buduće dobro oponašati zadanu funkciju i za parove ulaz izlaz koji joj nisu prezentirani tijekom učenja.

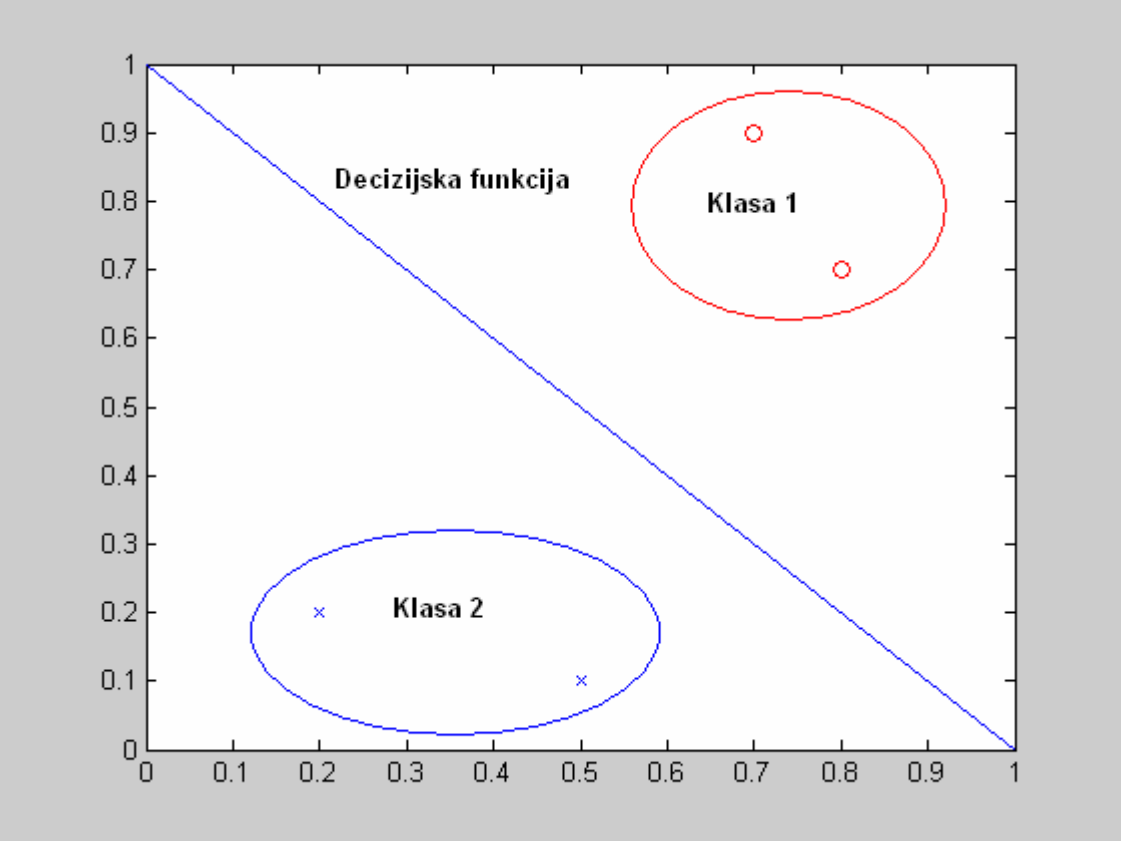
Postoji nekoliko grupa metoda za treniranje neuronskih mreža. U okviru ovog rada koristi se metoda povratne propagacije pogreške[5]. Nakon što neuronska mreža propagira ulazni vektor na izlazu iz mreže možemo očitati izlazni vektor. Ako je stvarna vrijednost izlaznog vektora poznata moguće je odrediti razliku između stvarne i dobivene vrijednosti. Dobivena razlika može se iskoristiti za korekciju izlaznog sloja neurona. Rezultat aktivacije izlaznog sloja neurona također se može promatrati kroz pogrešku, tj. razliku između željene i dobivene vrijednosti predzadnjeg sloja neurona. Dobivena razlika može se koristiti za korekciju predzadnjeg sloja neurona.

Iteriranjem navedenog postupka korekcija se provodi skroz do ulaznog sloja, a pogreška na izlazu se minimizira.

Jedna je od glavnih primjena neuronskih mreža klasifikacija ulaznog uzorka. Kada ulazni sloj ima broj neurona jednak vektoru značajki, a izlazni sloj broju klasa, tada se neuronska mreža može koristiti kao klasifikator. Neuron u izlaznom sloju koji ima najveću aktivaciju predstavlja klasu ulaznog uzorka.

Neuronska mreža uz dovoljan broj neurona i slojeva može učinkovito klasificirati i nelinearno separabilne uzorke, zbog nelinerarne karakteristike funkcije koju aproksimira.

[5] Eng. Backpropagation algortihm



Slika 3. Funkcija odlučivanja koja razdvaja dvije linearni separabilne klase

Na slici 3. možemo vidjeti dvije linearno separabilne klase u dvodimenzionalnom prostoru. Plavi križići označavaju vektore jedne klase, crveni kružići vektore druge klase. Plavi je pravac graf linearne funkcije odlučivanja koja odvaja dvije klase.

Jedna od glavnih prednosti neuronskih mreža u klasifikaciji naspram drugih metoda je vrlo visok stupanj generalizacije. Generalizacijom će neuronska mreže aproksimirati i one dijelove funkcije za koje nije dobila primjere tijekom učenja. U okviru klasifikacije to znači da će neuronska mreža biti u stanju ispravni klasificirati i one uzorke koji nisu korišteni tijekom treniranja i prvi put se klasificiraju.

Neuronske mreže se također u praksi pokazuju kao dobar pristup rješavanju različitih NP problema kao što su predviđanje i modeliranje podataka, klasifikacija uzorka, donošenje odluka, data miningu, sl.

2.3. Inteligentni sustav zasnovan na neuronskoj mreži

Obrada podataka u neinteligentnim sustavu sastoji se od transformacije ulaznih podataka čvrstim i nepromjenjivim algoritmom u izlazne podatke.

Inteligentni sustav u obradu podataka uvodu komponentu znanja. Znanje je skup pravila i podataka stečenih nekim oblikom učenja, a koji sada zajedno s čvrstim algoritmom transformiraju podatke. Znanje najčešće proširuje mogućnosti algoritamske obrade i mijenja sami smisao programa.

S matematičkog stajališta, pod transformacijom podataka podrazumijevamo matematičku funkciju. Složenom matematičkom funkcijom možemo dakle opisati algoritamsku transformaciju podataka, ali i stečeno znanje ako u nju uvedemo slobodne parametre.

Obzirom da uglavnom ne znamo eksplicitni oblik funkcije koju tražimo, u praksi se često koriste aproksimacije. Tu se neuronska mreža ističe kao naročito prilagodljiv sustav koji za aproksimaciju koristi poznate parove ulaz-izlaz, te zbog svojstva generalizacije dobro opisuje traženu funkciju.

3. Simulirani višerobotski sustav i robotski nogomet

Za istraživanje, razvoj i verifikaciju modela inteligentnog sustava potrebno je kao preduvjet imati i prikladnu platformu.
Robotski nogomet definira jednostavan standardni problem, za čije se rješavanje može koristiti širok skup različitih rješenja, a obuhvaća prikupljanje i analizu podataka, upravljanje robotima, te planiranje i izradu strategije.
Prikupljanje podataka provodi se kroz senzore, u okviru ovog rada, kroz kameru instaliranu iznad igrališta. Slika prolazi kroz obradu gdje se određuje pozicija svakog robota i loptice, te na temelju prethodnih podataka brzina i kutna brzina.
Sučeljem prema robotima realizira se upravljanje, gdje se zadavanjem linearne i kutne brzine realizira kretanje robota. U okviru ovog rada pod upravljanjem se podrazumijevaju i naprednije funkcije kao što je automatsko kretanje po nelinearnim krivuljama.
Ovaj rad se fokusira na treći aspekt problema robotskog nogometa a to je planiranje strategije i koordinaciju robota u svrhu povećanja uspjeha u natjecanjima.

Pravila robotskog nogometa su jednostavna. Dva protivnička tima po pet robota, kreću iz startnih pozicija i pokušavaju dati gol protivničkoj momčadi. Kada loptica prijeđe gol liniju nekog tima, protivnički tim dobiva bod i utakmica kreće iz startnih pozicija. Tim koji nakon 10 minuta igre ima više bodova je pobjednički tim.

Za izradu ovog diplomskog rada koristi se simulator koji je razvio ACT-Croatia tim na FERu. Simulator je napisan u C++ programskom jeziku i implementira osnovne funkcije potrebne za razvoj modela upravljanja, koje se modularno nasljeđuju po principima objektno-orijentirane paradigme. Važno je istaknuti da sam simulator prema korisniku osim sučelja ima i prikaz 3D simulirane okoline ODE standardom[6].

[6] ODE - eng. Open Dynamics Engine, otvoreni standard koji propisuje fiziku virtualne trodimenzionalne simulirane okoline, posebno prikladan za opis vozila, sudara tijela i nelinearnog gibanja. Standard uključuje i C/C++ biblioteke funkcija te je platformski neovisan.

Slika 4. Roboti i loptica za igranje

Simulator je izgrađen da simulira stvarnu ploču i stvarne robote. Stvarni sustav je opisan propozicijama i sastoji se od igraće ploče definiranih dimenzija[7], kamere iznad igrališta koja šalje sliku računalu na obradu[8], te samih robota. Roboti su oblika kocke dužine stranice od 7.5 cm, a pokreću ih dva elektromotora na koje su spojeni kotači. U verziji koja se koristi u okviru projekta na FERu, roboti udarac lopte postižu sudarom sa loptom, iako postoji i verzija sa posebnim elementom za udarac. Upravljački signali robotima se šalju bežičnim sustavom.

[7] Postoji nekoliko kategorija igračkih ploča i njihove dimenzije ovise o propozicijama pojedinih natjecanja.

[8] Glavni senzorski sustav. Pomoću slike dobivene sa igrališta se određuju pozicije svakog robota, a na temelju ranijih podataka i njihove linearne i kutne brzine.

4. Upravljanje višerobotskim sustavom zasnovano na neuronskim mrežama

4.1. Koncept

Sustav utemeljen na ako-onda pravilima izgrađuje se na temelju ljudskog opažanja i iskustva. Takav sustav za iste parametre uvijek donosi iste odluke i daje iste odgovore, pa je i njegov osnovni nedostatak neotpornost na promjene i nemogućnost poboljšavanja performansi.

Inteligentni sustavi kao ključnu komponentu kod donošenja odluka koriste znanje. Znanje im omogućava prilagodbu na trenutno stanje okoline, te omogućuje davanje optimalnog odgovora u okviru znanja kojeg posjeduje.

Prirodno se nameće činjenica da kada bi imali kvalitetnije znanje, i odluke koje bi se donosile bi maksimizirale šanse za postizanje cilja.

Problem koji nastaje u ovom pristupu je kako sakupiti znanje koje će se koristiti u donošenju odluka, te kako ga dalje proširivati i otkrivati novo znanje.

Ovaj rad predlaže model u kojem računalni sustav samostalno metodom pokušaja i pogrešaka pretražuje prostor mogućih odluka. Vodeći se činjenicom da različite sekvence odluka u konačnici daju različit uspjeh, sustav će pokušati u prostoru svih mogućih sekvenci usvojiti one najuspješnije.

4.2. Problem odlučivanja

Problem odlučivanja možemo svesti na problem klasifikacije, gdje ispravna klasifikacija ujedno znači ispravnu odluku.

Ulazni vektor je točka u konačnom prostoru svih mogućih stanja na igraćoj ploči, te se svakoj od tih točaka pridružuje odluka tj. klasifikacija.

U tako predstavljenom problemu potrebno je samo odrediti optimalnu funkciju odlučivanja klasifikatora[9], znači izgraditi sustav koji će svaki ulazni vektor ispravno klasificirati.

Obzirom da unaprijed ne znamo optimalnu funkciju odlučivanja, možemo je jedino s većim ili manjim uspjehom aproksimirati.

Neuronska mreža kao složena nelinearna funkcija, utvrđenim metodama treniranja i uz dovoljan skup primjera za učenje može vrlo dobro aproksimirati takvu zamišljenu funkciju.

[9] U okviru ovog razmatranja zanemaruje se vremenska domena koja prostor ulaznih vektora proširuje za još jednu dimenziju i na taj način ga čini beskonačnim. Primjer takvog slučaja je odluka utemeljena na proizvoljnom broju prethodnih odluka i stanja. Uzimajući u obzir navedenu činjenicu logično je zaključiti da optimalna funkcija odlučivanja ne može postojati.

4.3. Vrednovanje odluke i pretraživanje prostora mogućih odluka

Simulator u diskretnim vremenskim trenutcima za ulazni vektor s diskretnim vrijednostima donosi diskretne odluke, što u teorijskom razmatranju prostor niza mogućih odluka za sve moguće ulazne vektore čini konačnim. Takav konačan prostor je s matematičkog stajališta moguće u potpunosti evaluirati, i svakom nizu mogućih ulaznih vektora pridružiti optimalan niz odluka.
To je dakako u praksi neizvedivo, i ovakvim problemima se pristupa na razne načine.

Niz odluka može se evaluirati njegovim ukupnim uspjehom u postizanju cilja, ali se također mogu promatrati i pojedini podnizovi u okviru postizanja međurezultata. Međurezultat kad posljedica podniza odluka može biti dobar iako je ukupni niz odluka neuspješan u dolasku do cilja. Takav učinkovit podniz odluka treba usvojiti i koristiti za buduće odlučivanje, dok neučinkoviti ostatak treba izmijeniti.

Traženje novog niza odluka koji će biti uspješniji od prethodnog nije trivijalan problem. Obzirom da je niz odluka neposredna posljedica funkcije odlučivanja koju uz to još i ne znamo, traženje optimuma takve funkcije nije moguće riješiti analitičkim metodama.
Metaheuristički[10] pristup poput tzv. simuliranog kaljenja[11] daje moguće rješenje ovakvog problema. Iako ne postoji garancija da će ova i slična metoda ikada pronaći globalni optimum, iteriranjem se postupno pretražuje domena funkcije i pronalaze lokalni optimumi.

Blagim promjenama niza odluka za neko stanje okoline, možemo djelomično promijeniti i ukupne performanse sustava. Poboljšanje performansi će se usvojiti a pogoršanje odbaciti i nastaviti sa daljnjim promjenama niza odluka.

Takve promjene u sustavu odlučivanja zapravo su metoda pretraživanja prostora svih mogućih nizova odluka, na čemu se temelji sustav otkrivanja i sakupljanja novog znanja opisan u ovom radu.

---

[10] Metaheuristika – je neka općenita metoda koja služi rješavanju široke klase računskih problema, pri tome se koristeći nekim intuitivnim pravilom u nadi da će se rezultat s vremenom poboljšavati.
[11] Metaheuristička metoda traženja lokalnog optimuma nadahnuta metalurškim procesom kontroliranog hlađenja čelika, u svrhu stvaranja pravilnije kristalne strukture.

4.4. Osnovna strategija i robotske uloge

Moderna teorija odlučivanja[12] dijeli odluke po vrsti na: strateške, taktičke i operativne. Strateške odluke oblikuju dugoročni cilj, apstraktnog su oblika i smjernica su nižim odlukama. Taktičke odluke služe realizaciji strateških odluka, oblikuju stvarne događaje i vode se aktualnim podacima i iskustvom. Operativne odluke su programirane rutine i donose se po unaprijed utvrđenim obrascima.

Slika 5. Piramida odlučivanja

Na tragu ovom modelu oblikovan je i sustav za odlučivanje opisan u ovom radu. Odlučivanje unutar sustava se odvija na dvije razine. U odnosu na referentni teorijski model mogli bi reći da se u jednoj razini donose strateške i neke taktičke odluke, dok se u drugoj razini donose operativne i uglavnom niže taktičke odluke.
Takav raspored proizlazi iz modela upravljanja robotima zasnovan na ulogama u igri.

U igri postoji pet mogućih uloga: golman, obrambeni igrač, vezni igrač, napadač i centarfor.

Svaki robot, osim golmana, može prihvatiti bilo koju ulogu u igri u bilo kojem trenutku, dok je golmanova uloga fiksirana i nepromjenjiva.

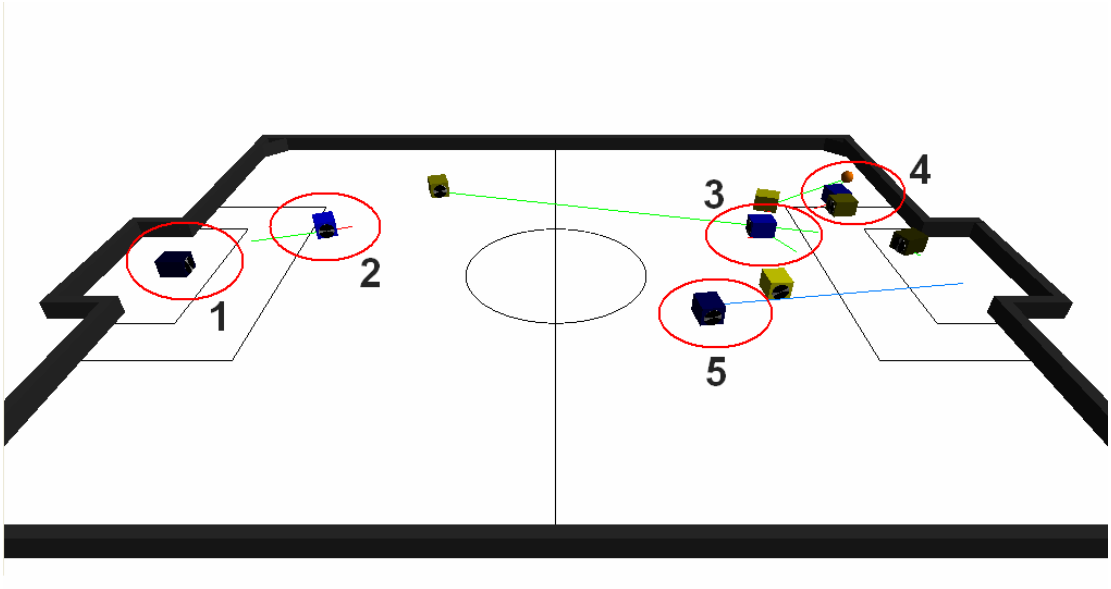
[12] Navedeni model se odnosi na strukturu odlučivanja unutar kompanije ili radne skupine sa više članova

Dodjela uloga svakom robotu u svakom trenutku simulacije donosi se bez utjecaja na autonomno ponašanje robota nakon dodijele uloga. Viša razina odlučivanja tako dodjelom uloga realizira strateške i djelomično taktičke uloge. Ako se na višoj razini odlučivanja procijeni da je vrijeme za napadački pristup utakmici, tada se robotima dodijele napadačke uloge, te se utakmica odvija u napadačkom tonu prema protivniku. Ako se procjeni da je vrijeme za obrambeni pristup, uloge koje se sada dodjeljuju su obrambenog karaktera.

Rasporedom i brojem dodijeljenih uloga među robotima stvara se i taktička pozicija prema protivničkom timu. Ako se npr. sustav odluči za napadačku ulogu tima i dodijeli puno napadačkih uloga svojem timu, roboti će agresivnije pristupiti probijanju protivničke obrane i guranju loptice prema protivničkom golu. Time se povećava šansa za proboj prema protivničkom golu, ali također i povećava mogućnost realizacije kontranapada protivničkog tima.

Dakle kritični dio uspješne strategije je pravilna i pravodobna dodjela uloga. U osnovnoj verziji strategije dodjela uloga vrši se ako-onda pravilima.

Same uloge svakog robota većim su dijelom automatizirane i trebaju malo parametara za rad. Ovisno o ulozi programira se ponašanje robota u dva sloja. Kretanje, rotacija i korekcija brzina, te izbjegavanje sudara programirani su u prvom sloju. Automatskom regulacijom brzine vrtnje elektromotora postižu se svi oblici kretanja. U drugom sloju programirano je ponašanje robota elementarnijim funkcijama iz prvog sloja.



Slika 6. Crveno zaokruženi roboti su numerirani svojim ulogama

1. Uloga golmana realizira obrambenog igrača na golu. Prateći y-poziciju loptice nastoji spriječiti njezin ulazak u prostor gola. Ako se loptica nalazi u njegovoj blizini pokušat će ju izgurati u teren dalje od gola.

2. Obrambeni igrač pokušava spriječiti napredovanje loptice prema golu stvarajući prepreku loptici ispred gola. Ako postoji više obrambenih igrača raspoređuju se u sprječavanju loptice i u pokrivanju protivničkih igrača u blizini. Raspored maksimalno prekriva dužinu gola.

3. Vezni igrač pokušava spriječiti napredovanje loptice i istovremeno je pokušava odgurati prema protivničkom golu. Prateći poziciju i kretnje loptice izračunava mjesto sudara i trajektoriju kretanja.

4. Napadač pokušava udarcima lopticu gurnuti u protivnički gol ili u blizinu protivničkog gola. Njegovo je kretanje nešto brže a krivulje kretanja su prilagođene optimalnom kutu udarca.

5. Centarfor je uloga namijenjena jedino davanju gola protivničkom timu. Robot traži najbolju poziciju za udarac tražeći pravac prema golu na kojem nema prepreka.

Predviđa[13] vrijeme dolaska loptice u kretanju na njegovu liniju udarca, te ravnomjerno ubrzava do udarca pokušavajući ugurati lopticu u gol.

4.5. Nadogradnja neuronskom mrežom

Ljudski programer će u razvoj sustava za strategiju robotskog nogometa uložiti konačno vrijeme. Računalo za razliku od čovjeka nije ograničeno umorom, emocijama i subjektivnošću, međutim nije niti sposobno simulirati intuiciju i inovaciju ljudskog intelekta.

Međutim, vrlo visoka računalna moć današnjih računala i razvijeni matematički modeli otvaraju mogućnosti učinkovite pretrage svih mogućih načina dostizanja nekog cilja. Računalo nikada neće izaći iz okvira prostora problema koji pretražuje, ali će uz dano dovoljno vremena vrlo vjerojatno u tom prostoru pronaći bolje rješenje od čovjeka.

Funkcija odlučivanja klasifikatora dobrih i loših odluka na temelju prikupljenih podataka sa igrače ploče je jedno takvo rješenje u prostoru problema. Čovjek oblikuje sustav za odlučivanje ako-onda pravilima ili parametriziranjem funkcije, na temelju iskustva i intuicije.

Neuronska mreža je u kontekstu spomenutog klasifikatora upravo ta funkcija odlučivanja, a pretpostavka izgradnje takve mreže je da će odluke koje se pokažu dobre za neko stanje na igraćoj ploči, također biti ispravne i za drugo slično stanje.

Ispravna sekvenca odluka tj. klasifikacija uzima se kao uzorak za treniranje neuronske mreže. Zbog svojstava generalizacije, neuronska mreža će i u budućnosti slične primjere klasificirati sličnim vrijednostima. Pretraživanje prostora svih mogućih rješenja postupno dovodi do sve boljih rezultata pri dodjeli uloga i krajnjem uspjehu u natjecanju.

Dodjela uloga ako-onda pravilima u osnovnoj verziji strategije, zamjenjuje se u potpunosti neuronskom mrežom.

[13] Predviđanje kretanja loptice također je riješeno neuronskim mrežama

Ulaz u neuronsku mrežu je vektor stanja sa igrače ploče. On je skup svih mjerenja i obrađenih podataka dostupnih sa igraće ploče. Neuronska mreža propagacijom signala vrši obradu primljenih podataka i odlučuje o podjeli uloga. Izlaz neuronske mreže je vektor u kojem se svakom robotu numerirano dodjeljuje uloga.

Nakon odigrane utakmice, analiziraju se rezultati i međurezultati pojedinih dijelova utakmice pa tako i odluke koje su dovele do njih. Nakon analize igre priprema se skup za učenje.
Skup za učenje su parovi vektora od kojih je jedan ulazni vektor a drugi traženi izlazni vektor koji mreža treba naučiti.

Ulazni vektori su spremljena stanja na igraćoj ploči u svakom trenutku iz prethodne utakmice.
Izlazni vektori u skupu za učenje i njihova korekcija u odnosu na odgovor mreže za pripadni ulazni vektor, su osnovni mehanizam djelovanja na mrežu sustava za treniranje mreže.

U ranijem tekstu navodi se činjenica da optimalno ponašanje sustava za odlučivanje nije unaprijed poznato, pa tako niti potpuno ispravni odgovori nisu unaprijed poznati.

Sustav na temelju heurističkih pravila određuje hoće li se izlazni vektor za pripadni ulazni vektor mijenjati ili će ostati isti. Ako ostane isti, neuronska mreža će uz još veću vjerojatnost sljedeći put za slični ulazni vektor odgovoriti istim odgovorom, dok će kod promijenjenog izlaznog vektora za učenje odgovor težiti ka novom vektoru.

Modifikacija i odabir izlaznih vektora ima ključnu ulogu u napredovanju performansi sustava. Proces odabira mora iskoristiti osobine sekvenci dobrih odluka i istovremeno održati mogućnost napretka kroz pretraživanje.

Evaluacija podniza odluka mora zadržati visoki stupanj općenitosti, kako bi međuzavisnost pretrage i ocjene ostala što manja. Pretraga koja je pod visokim utjecajem ocjene zanemaruje neke mogućnosti, tj. ne pretražuje cijeli prostor mogućih rješenja.

Traženjem novih rješenja sustav ostvaruje poboljšanje performansi, što bi cijeli proces moglo okarakterizirati kao učenje.

Najveća prednost ovakvog automatskog učenja je u tome da ne traži daljnje uplitanje čovjeka. Samostalnost i učenje iz primjera otvara mogućnost da će neuronska mreža tijekom vremena pronaći skrivene međuzavisnosti nevidljive ljudskom programeru.
Samostalno stjecanje znanja bez potrebe eksperta također bitno ubrzava napredak sustava, koji sada ovisi jedino o brzini računala.

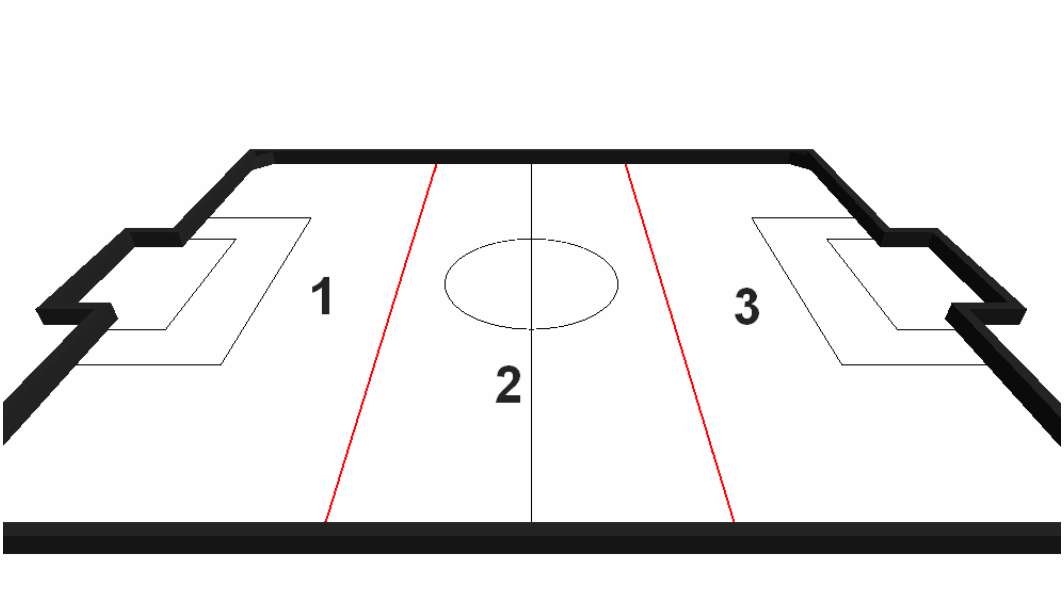
4.6. Algoritam učenja strategije

Učenje strategije provodi se nakon odigrane utakmice i sakupljenih rezultata. Sva mjerenja i sve pripadne odluke spremaju se u obliku matrice u memoriju računala.

Utakmica može završiti pobjedom, porazom ili istekom vremena. I dok pobjeda ili poraz daju jasnu ocjenu uspjeha postizanja cilja, neriješen rezultat traži podrobniju analizu igre i napretka kroz igru.
Međutim čak i kada igra završi pobjedom, ne znači da su svi njeni dijelovi bili jednako uspješni, a isto vrijedi i za igru koja završi porazom.

Evaluacija podniza odluka mora se moći ocijeniti trenutnim napretkom u odnosu na ukupnu igru. Da bi se zadržao visok stupanj općenitosti prilikom odlučivanja referentna točka koja će se promatrati u svakom trenutku je pozicija loptice.

U odnosu na kretanje i poziciju loptice vrši se ocjena sekvence odluka. Sekvence odluka se izdvajaju na temelju pozicije loptice na igraćoj ploči i u svrhu toga ploča se dijeli po dužini na tri jednaka dijela: obrambeni, vezni i napadački.



Slika 7. Tri regije na igraćoj ploči.

Svaki od tri dijela ploče ima različit kriterij uspjeha i na kraju ocjene.

Obrana u igri ima za cilj spriječiti protivničku momčad da dođe do gola. Podjela obrambenih uloga će u pravilu spriječiti napredovanje protivničke momčadi. Međutim, što se dulje loptica zadržava u okolini prijateljskog gola, to se povećava šansa za zgoditak protivnika. Dakle, obrana koja uspije čim prije poslati lopticu prema protivničkom golu je uspješna obrana.
Obrana se evaluira prema dva kriterija: ne primanje gola, i što kraće zadržavanje loptice u prvoj trećini terena u odnosu na ukupno trajanje utakmice.

U veznom dijelu terena se najčešće pokušava spriječiti napredovanje protivnika i pokušava realizirati proboj prema protivničkom golu. Da bi se zadržao kriterij općenitosti, evaluacija srednjeg dijela terena neće biti pod utjecajem pobjede ili gubitka u utakmici.
Promatraju se parovi pozicija loptice u trenutku n-1 i n. Ako je vrijednost y-koordinate povećana u trenutku n u odnosu na trenutan n-1 tada je loptica napredovala prema protivničkom golu i to je pozitivni pomak. U obrnutom slučaju je pomak negativan. Ako prebrojimo sve pozitivne i negativne parove za neki period trajanja utakmice, ocjena sekvence se može dobiti kao omjer ta dva broja.

Treći, protivnički dio terena, je mjesto s kojeg će se pokušati postići zgoditak. Ako podjela uloga u protivničkom terenu rezultira pobjedom, sekvenca odluka će se ocijeniti visokom ocjenom. Međutim, i dulje zadržavanje loptice u tom dijelu terena će povećati šanse za eventualan zgoditak, pa će se ocjena utakmice koja nije završila pobjedom evaluirati na temelju vremena provedenog u protivničkom dijelu terena. Također srednja vrijednost pozicije lopte, tj. udaljenost od protivničkog gola igra ulogu u formiranju ocjene.

Sve tri ocjene nastoje smanjiti međusobnu zavisnost i omogućiti kreiranje parametara kojima će se oblikovati skup za učenje.

4.7. Predviđanje kretanja loptice

Uloga centarfor predviđa samo pokušaj davanja gola. Dok je učinkovitu obranu relativno jednostavno izvesti, davanje gola nije trivijalan problem. Robot se u ulozi centarfora postavlja u prikladnu poziciju i čeka nailazak loptice na svoju liniju udarca.

Kretanje loptice analitički se može promatrati kroz nekoliko diskretnih vremenskih trenutaka te se trajektorija može interpolirati iz dvije ili više točaka. Međutim, situacija u kojoj se nalazi približno 8[14] robota prilično je kaotična te se učinkovitost predviđanja kretanja loptice smanjuje s povećanjem broja robota u blizini loptice.

Jedna od često korištenih primjena neuronskih mreža je predviđanje. Promatrano s teoretske strane, svaki je proces određen konačnim brojem parametara[15]. Kada bismo imali na raspolaganju apsolutno sve parametre, ponašanje procesa bilo bi potpuno moguće predvidjeti i opisati funkcijom. Kako svi faktori nisu na raspolaganju a proces je u naravi stohastički, poznate elemente ćemo koristiti u proračunu, a sav ostali utjecaj ćemo aproksimirati šumom.

Neuronska mreža može dobro aproksimirati funkciju koja bi opisivala ponašanje sustava ili promjenu varijable kroz vrijeme. Skup za učenje formira se iz podataka sakupljenih o ponašanju funkcije u prošlosti, u nadi da će aproksimacija funkcije iz prošlosti dobro opisivati i ponašanje funkcije u budućnosti.

[14] Oko 5 protivničkih i 3-4 prijateljska robota u blizini gola.
[15] Teza teorije determinističkog kaosa.

Za ulogu centarfor formira se posebna neuronska mreža kojoj je jedini posao predviđanje kretanja loptice u budućnosti.
Neuronska mreža predviđa položaj loptice pet vremenskih koraka[16] u budućnosti, a kao ulaz uzima vektor stanja na igraćoj ploči. Skup za učenje se formira iz stanja spremljenih tijekom igre.

Predviđena pozicija nakon pet vremenskih koraka daje bolju interpolaciju trajektorije loptice i bolje određivanje točke udarca robota.

[16] Vrijeme u simulaciji je diskretizirano na fiksne vremenske korake, unutar kojih se odvijaju proračuni, donose odluke i realizira upravljanje.

5. Implementacija

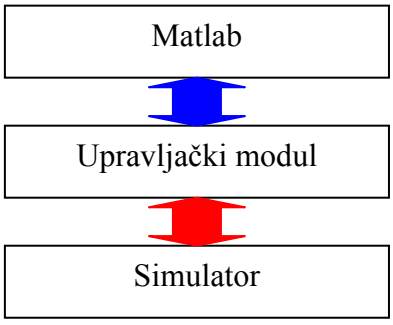
5.1. Simulator- Upravljački modul - Matlab arhitektura

Senzorski je sustav zasnovan na digitalnoj kameri postavljenoj iznad igrališta. Na svakom robotu u igri nacrtan je geometrijski simbol koji ga jednoznačno određuje. Obradom slike može se utvrditi točna pozicija svakog robota i loptice u koordinatnom sustavu igrališta.

Simulator na raspolaganju ima simulirane podatke, dakle pozicije se ne određuju analizom digitalne slike, nego ih simulator generira.

Modul za upravljanje implementiran je nasljeđivanjem CSoccerStrategy bazne klase simulatora. U baznoj klasi simulatora implementirana je sva elementarna funkcionalnost, kao što je slanje upravljačkih signala robotima i pristup obrađenim podacima senzora. Osim elementarnih, implementirane su i neke naprednije funkcije kao što je kretanje prema zadanoj točki i izbjegavanje sudara.

Promatrajući arhitekturu sustava iz perspektive upravljanja strategijom, čitav se sustav može podijeliti na tri sloja.



Slika 8. Arhitektura sustava za upravljanje robotskim nogometom

Između slojeva postoji dvosmjerna komunikacija, a svaki od slojeva provodi određeni dio ukupne obrade podataka.

Simulator je najniži sloj sustava za upravljanje. U njemu je implementirana obrada podataka prikupljenih senzorima, te njihov prijenos u više slojeve. Simulator od viših slojeva prima upravljačke signale na simboličkoj razini, kao što je linearna i kutna brzina, te ih obrađuje i izvršava u obliku stvarnog kretanja robota.

Upravljački modul izvršava većinu algoritamske obrade podataka. Nakon što od simulatora primi mjerenja, upravljački modul mora analizirati situaciju i pridijeliti svakom robotu neku ulogu. Unutar algoritamske obrade svake uloge nalazi se funkcionalnost koja određuje ponašanje svake uloge, proračun kretanja robota, te proračun linearne i kutne brzine za zadano kretanje.
Strategija i dodjela uloga kao posljedica strategije izračunavaju se u sloju iznad upravljačkog modula, te on formira vektor stanja koji proslijedi Matlab sloju. Iz njega dobiva vektor koji sadrži numerirane uloge, koje sada upravljački modul dekodira i izvršava na pripadnim robotima.
Matlab[17] sloj je najviši sloj i kao takav sadrži i najviše funkcije sustava za upravljanje. U njemu su implementirane neuronske mreže, dodjela uloga te sustav za analizu i treniranje neuronske mreže.

Raslojavanjem sustava postiže se modularnost i neovisnost unutarnje implementacije.

Osnovna karakteristika svakog inteligentnog sustava vidljiva je kroz protok podataka simulator-upravljački modul-simulator, dok je u upravljačkom modulu realizirana algoritamska obrada, te znanje spremljeno u Matlab sloju.

5.3. Upravljački modul

Upravljački modul realiziran je klasom koja se zove CMatblabStartegy, a nasljeđuje klasu CSoccerStartegy. Konstruktor klase inicijalizira Matlab server i u njemu pokreće skriptu «InicijalizirajWorkspace.m» koja služi inicijalizaciji Matlab okoline.

[17] Matlab je matematičko-analitički računalni alat, sa implementacijama mnogih matematičkih modela pa tako i neuronskih mreža.

Glavna logika upravljanja smještena je u Strategy() metodu. Strategy metoda je apstraktna metoda baznog razreda te se mora implementirati zasebno. Simulator metodu poziva jednom za svaki diskretni vremenski trenutak simulacije.

Pseudokod:

```
ako je došlo do gola ili je vrijeme simulacije isteklo
onda
{
	pokreni Matlab skriptu za treniranje neuronskih mreža
(PodjelaUlogaUcenje.m);
	resetiraj simulaciju;
}
inače
{
	pripremi vektor stanja sa podacima iz igre;
	pozovi Matlab skriptu za podjelu uloga (PodjelaUloga.m);
	pročitaj vektor odluka sa Matlab servera;
	dekodiraj uloge i pozovi pripadne metode za pripadne robote;
}
```

Matlab kao sustav ima implementiranu C/C++ biblioteku funkcija, tako da se pozivanje skripti i predavanje parametara Matlabu izvodi metodama na engineu Matlaba. Engine Matlaba je u stvari referenca na Matlab server pokrenut na računalu.

Vektor stanja je dimenzije 29 i sastoji se od koordinatnih pozicija svih robota u igri, dosadašnjih uloga prijateljskih robota, te pozicije brzine i kutne brzine loptice. Mjerenja i pozicije dobivaju se iz podataka simulatora, a vektor se formira pozicijski, što znači da su isti parametri istog robota uvijek na istome mjestu. Nakon što se vektor spremi u memoriju računala Matlabu se predaje pokazivač na vektor u memoriji.

Nakon što Matlab izvrši skriptu «PodjelaUloga», u memoriji se preko reference dolazi do vektora podjele uloga. Uloge su numerirane brojevima od 1 do 4, tj. svaki element vektora podjele uloga ima vrijednost od 1 do 4.
Uloga numerirana brojem 1, nakon dekodiranja odgovara ulozi obrambenog igrača, uloga numerirana brojem 2 odgovara ulozi veznog igrača. Dok uloge numerirane sa 3 i 4 odgovaraju ulogama napadača i centarfora.
Svaki je robot u simulaciji numeriran svojim brojem. Vektor podjele uloga je dimenzije 4 te se pozicijski svakom robotu pridjeljuje njegova dekodirana uloga. Svaki robot može izvršiti bilo koju ulogu, iako će se zbog naravi cijelog sustava neke uloge s većom vjerojatnosti dodjeljivati pojedinim robotima.

Interesantna je činjenica da roboti 2 i 4 češće dobivaju napadačke uloge, dok roboti 1 i 3 češće dobivaju obrambene uloge.

Robot numeriran brojem 0 uvijek ima funkciju golmana.

Maksimalno vrijem trajanja simulacije definira se varijablom MAX_ITERATION_NUM. Iako broj može biti proizvoljan, zbog memorijskih i procesnih ograničenja računala on je za većinu pokusa u okviru ovog rada definiran na 5000.

Vremenski korak definiran je parametrima simulacije. Ako je vremenski korak[18] malen, tada se procesiranje mora izvesti u kraćem vremenu, ali je i broj mjerenja i određivanja pozicija također veći. U okviru ovog rada on je uvijek konstantan i iznosi 12.5 ms.

Dimenzije igrališta su normirane u interval [0.0, 2.2] za dužinu i [0.0, 2.0] za širinu igrališta. Davanje ili primanje gola registrira se izlaskom x-koordinate loptice iz intervala za dužinu[19].

[18] Eng. sampling period – vrijeme uzorkovanja
[19] Mjerenja koordinata mogu ići i izvan zadanog intervala, kao i u negativne vrijednosti.

5.4. Matlab modul

Matlab je za potrebe ovog rada odabran zbog dobrih analitičkih svojstava kao i mnogih postojećih ugrađenih funkcija. Skriptni jezik na kojem se baziraju programi jednostavan je za izmjenu, a alati za vizualizaciju olakšavaju analizu.

Neuronska mreža se instancira definiranjem broja slojeva i broja neurona po svakome sloju, te aktivacijskom funkcijom. Interno unutar Matlaba neuronska mreža se pohranjuje u obliku matrica koje su sastavljene od težinskih vektora ili pak vektora pragova.

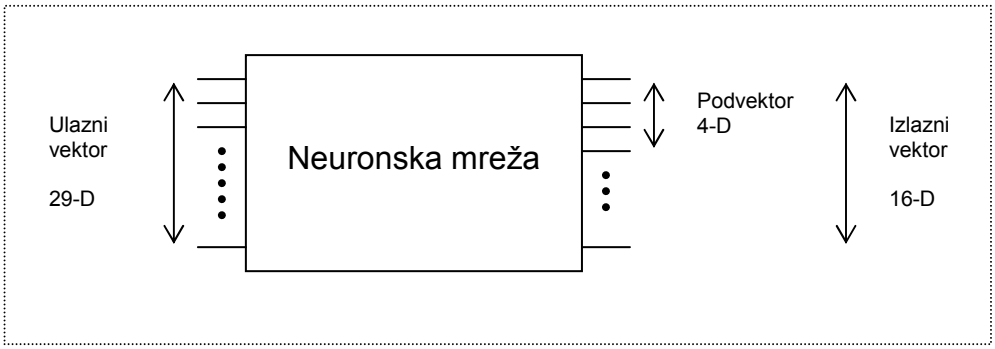
Matlab modul tijekom inicijalizacije učitava u memoriju matrice koje opisuju neuronsku mrežu iz datotečnog sustava. Tijekom inicijalizacije se formiraju i razne pomoćne strukture, međuspremnici i brojači. Nakon svakog treniranja neuronskih mreža i promjena na matricama, Matlab modul sprema promjene u datotečni sustav.
Za potrebe ovog projekta unutar Matlab modula napisane su skripte za inicijalizaciju neuronskih mreža i radne okoline, skripte za podjelu uloga, predviđanje kretanja loptice, te nekoliko pomoćnih skripti.

Implementaciju pojedinih uloga MatlabModul nasljeđuje, uz neke manje izmjene, iz osnovne verzije sustava za upravljanje robotskim nogometom. Promatranje utjecaja neuronske mreže na poboljšanje performansi tako se može još objektivnije obaviti, s obzirom da su moduli ravnopravni na razini automatskog upravljanja.

5.5. Neuronska mreža za podjelu uloga

Neuronska mreža za podjelu uloga sadrži 3 sloja. Prva dva sloja imaju po 30 neurona dok izlazni sloj ima 16 neurona, a svi neuroni koriste sigmoidnu aktivacijsku funkciju. U skladu s time svi elementi izlaznog vektora će biti u intervalu <0,1>.

Ulazni sloj prima 29-dimenzionalni ulazni vektor i prosljeđuje ga prvom sloju. Izlaz ni sloj od 16 neurona daje 16-dimenzionalni izlazni vektor.



Slika 9. Shema neuronske mreže

Izlazni vektor se dijeli na četiri podvektora tako da prve četiri vrijednosti čine prvi podvektor, druge četiri vrijednosti drugi podvektor i tako dalje. Svaki od tih podvektora daje odluku o dodijeli uloge za jednog robota.

Podvekotor je 4-dimenzionalan vektor, a uloga koja se dodijeli pripadnom robotu jednaka je poziciji najveće vrijednosti u podvektoru, tj. najjače aktivacije.

Tako da ako je npr. prvi izlazni podvektor jednak $[0.1, 0.2, 0.8, 0.2]^T$, uloga dodijeljena prvom robotu će biti uloga broj 3, zbog najjače aktivacije na trećoj poziciji. Pozicija broj 3 odgovara ulozi napadača pa bi u tom slučaju prvi robot preuzeo tu ulogu u upravljačkom modulu. Da je prva pozicija imala najjaču aktivaciju, prvi robot bi preuzeo ulogu obrambenog igrača itd.

Svaki podvektor daje jedan broj, što uzimajući u obzir da imamo četiri podvektora, daje četiri broja, tj. numerirane odluke. Tim brojevima kreiramo novi četverodimenzionalni vektor u kojem će svi elementi biti prirodni brojevi od 1 do 4. Ovaj vektor je će se u kasnijoj obradi predati upravljačkom modulu.

Ako promatramo izlaz neuronske mreže u cjelini, zapravo možemo reći da neuronska mreža daje procjenu svih uloga za sve robote, a da ona koja ima najbolju procjenu biva odabrana.

5.6. Dodjela uloga i «buffering»

Neuronska mreža u diskretnom trenutku simulacije svakom robotu dodjeljuje ulogu.

Može se dogoditi da dodjela uloga alternira između dviju vrijednosti, ili da tijekom prijelaza iz jedne u drugu ulogu dolazi do kratke sekvence stalnih promjena uloga..

Ponašanje robota u tom kratkom periodu postaje kaotično i nefunkcionalno, pogotovo kada se uzme u obzir da određene uloge imaju nekoliko faza, koje se stalnim promjenama uloga ne mogu realizirati.

Da bi se spriječio navedeni problem, za donošenje odluka se uvodi međuspremnik[20] odluka. Međuspremnik je zapravo matrica dimenzija 4x7 u koju se sprema posljednjih sedam vektora odluka.

U njega se sprema posljednjih sedam vektora, te svaki novi vektor odluka koji se pokuša ubaciti, istiskuje najstariji vektor. Međuspremnik da bi se svakom robotu dodijelila ona uloga koja se najčešće pojavljuje.

Ako npr. međuspremnik izgleda ovako:

3 4 4 3 3 3 3
2 2 2 2 1 2 2
4 4 4 4 4 2 2
2 2 3 3 3 3 3

vektor odluke koji će se proslijediti modulu za upravljanje biti će $[3\ 2\ 4\ 3]^T$.

5.7. Skup za učenje

Formiranje skupa za učenje ključan je dio u napredovanju sustava za donošenje odluka.

Skup za učenje formira se iz ulaznih vektora spremljenih tijekom utakmice i svih donesenih odluka za te vektore, koje su također spremljene u memoriju. Dakle sustav uči iz svojih prethodnih odluka.

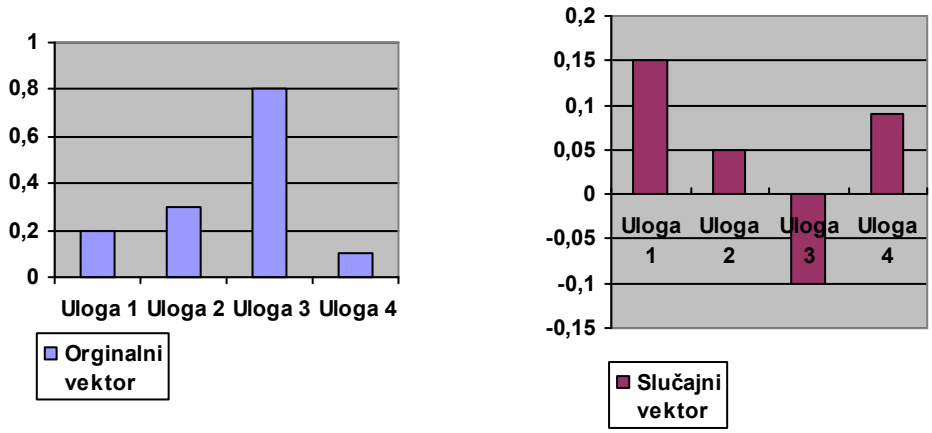
[20] Eng. buffer

Već je ranije spomenuta paradigma učenja kroz dijeljenje ulaznih i pripadnih izlaznih vektora ovisno o trećini terena na kojem se igra utakmica. Vektori se dakle dijele u tri skupine ovisno o x-koordinati loptice. Ako se loptica nalazi u prvoj, dakle obrambenoj trećini terena vektor se sprema u prvi skup, ako se nalazi u veznom dijelu terena onda se vektor sprema u drugi skup, te se treći skup popunjava vektorima iz protivničke trećine.

Svaki od skupova za učenje se modificira na poseban način, ovisno o ocjeni uspješnosti igre u tom dijelu terena.

Prethodno donesene odluke, ako se tako procijeni, se mogu koristiti prilikom treniranja. Međutim, ako se odluči da ranije donesena sekvenca odluka nije dala dobar rezultat, prijeći će se na modifikaciju tih odluka.
Da bi se zadržala općenitost i nesubjektivnost u eventualnoj promjeni odluka za neki ulazni vektor, svim mogućim ulogama treba dati jednaku šansu. Mogućnost da je i trenutna odluka zapravo dobra, ali krivo ocjenjena, treba se također odražavati preko metode promjene odluke.

Metoda opisana u ovom radu, uvodi tzv. šum u vektor odluka koji treba promijeniti. To znači da se na svaki vektor odluka, kakav je postoji u skupu za učenje, superponira slučajni vektor. On je iste dimenzionalnosti kao vektor odluke.

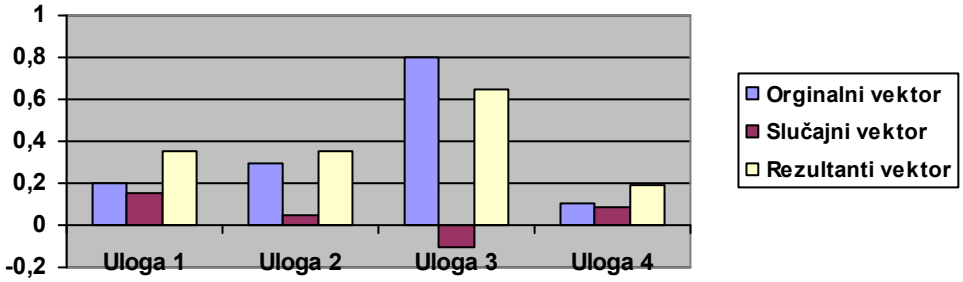


Slika 10. Primjer podvektora dodjele uloga i slučajni vektor

Slučajni je vektor modificiran na način da je pozicija elementa s najvećom aktivacijom podvektora za dodjelu uloga, pretvorena u negativnu vrijednost, a ostali elementi ostaju kao pozitivni brojevi. Zbrajanjem ovakvog slučajnog vektora s podvektorom odluka, oslabljuje se mogućnost da dosadašnja odluka u sljedećoj sličnoj situaciji opet bude donesena, a pojačava se mogućnost da sljedeći put dođe do neke druge odluke.

Na taj se način postiže mogućnost promjene taktike igre preko promjene uloga. Sustav sam, metodom pokušaja i pogrešaka, pretražuje sve moguće odluke za danu situaciju, dajući prednost odlukama koje su se dosada pokazala ispravnima, tj. koje su dobile visoku ocjenu.

Bitno je također napomenuti da superpozicijom slučajnog vektora, nismo eliminirali mogućnost ponovnog donošenja prethodne odluke, nego smo joj samo smanjili vjerojatnost pojavljivanja.



Slika 11. Rezultantni vektor

Na slici 11. možemo vidjeti utjecaj slučajnog vektora na rezultantni vektor.

Formula kojom se korigira stari izlazni vektor je:

$$\vec{n} = abs(\vec{s} - k * (\frac{1}{3} * (2 * \vec{sz} - rand(w,h))))$$

$\vec{n}$ – novi vektor

$\vec{s}$ – stari vektor

k – koeficijent korekcije

$\overrightarrow{sz}$ – stari zaokruženi vektor, gdje je najjača aktivacija zaokružena na 1 a ostala na 0

w, h – dimenzije izlaznog vektora (16x1)

abs – funkcija koja vraća apsolutnu vrijednost vektora

$rand$ – funkcija koja vraća vektor slučajnih brojeva po uniformnoj razdiobi

Iz formule je moguće primijetiti da se uniformna razdioba slučajnih brojeva od <0, 1> normira na način da ukupna srednja promjena teži u nulu.

U formuli se nalazi i koeficijent korekcije *k*. Taj broj u sebi sadrži dvije komponente. Jedna je ukupni utjecaj slučajnog vektora na stari izlazni vektor. Ako je koeficijent velik i bliži 1, tada je utjecaj slučajnog vektora jako velik. Smanjivanjem koeficijenta k prem nuli, smanjuje se i utjecaj slučajnog vektora.
Druga je komponenta regulacija srednje vrijednosti skupa izlaznih vektora. S obzirom da je interval u kojem se treba nalaziti svaki element izlaznog vektora neuronske mreže <0,1> i da se nastoji spriječiti dominacija jednog izlaza podvektora, srednja vrijednost svih izlaza mora težiti u 0.5.
Naime, djelomično slučajnim procesima, a djelomično zbog samog sustava za korekciju vektora, ukupna srednja vrijednost izlaza neuronske mreže može nakon mnogo iteracija učenja jako smanjiti vrijednost i težiti nuli. Također je moguć i obratni slučaj gdje srednja vrijednost teži ka 1. Tada je najjača aktivacija jako podložna šumu i slučajnim vrijednostima ukomponiranim u mjerenja, te sustav postaje neučinkovit.

Da bi se izbjegao navedeni scenarij, prije učenja se skup korigira apsolutnim koeficijentom korekcije. Taj broj povećava ili smanjuje srednju vrijednost ovisno o ukupnoj srednjoj vrijednosti skupa za učenje, obrnuto proporcionalno razlici srednje vrijednosti i očekivane srednje vrijednosti (0.5).

5.7. Treniranje neuronske mreže za podjelu uloga

Kako je već navedeno treniranje se provodi sa tri skupa za učenje.

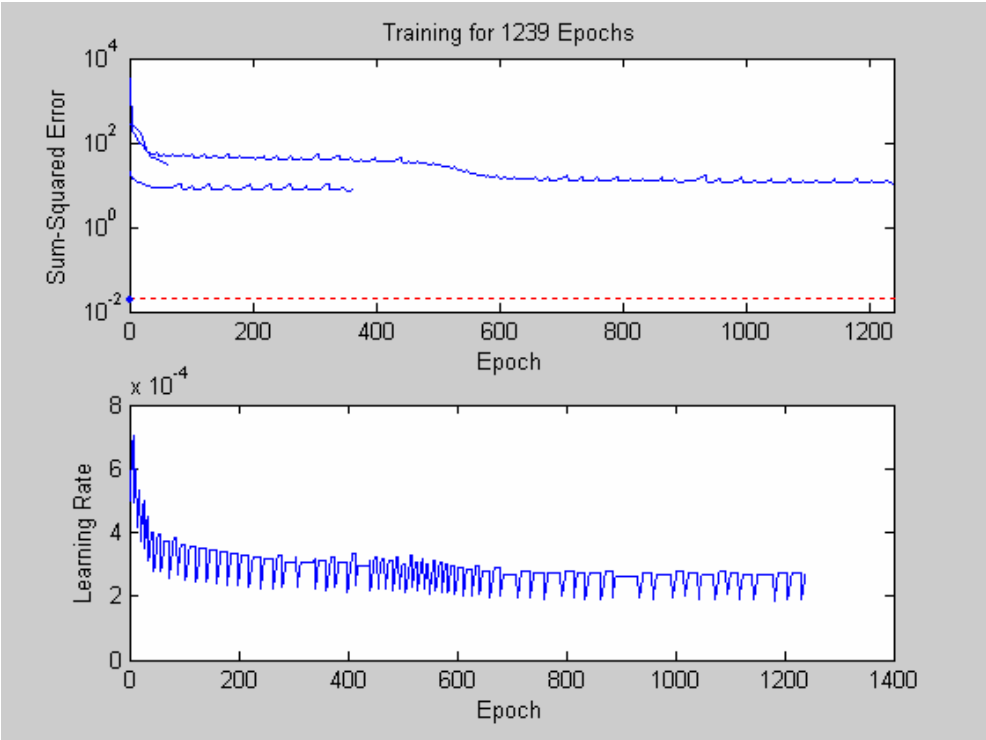
Skup za učenje u prvoj, obrambenoj, trećini igrališta formira se po kriteriju zadržavanja loptice u prvoj trećini igrališta u odnosu na ukupno trajanje utakmice, i po kriteriju dobivanja gola.
Ako je utakmica izgubljena, znači da je obrana loše odigrala utakmicu i da je potrebno napraviti veću korekciju. Već ranije spomenuti faktor korekcije se pojačava i skup za učenje se pojačano mijenja u odnosu na originalni izlaz.
Ako utakmica nije izgubljena tada se obrana evaluira na temelju omjera vremena provedenog u obrani i ukupnog vremena u utakmici. Ako je taj broj manji od 1/3 tj. ako je manje od jedne trećine bilo potrebno braniti gol, obrana se smatra uspješnom i korekcija se ne provodi, nego je skup za učenje jednak starim odlukama.

Skup za učenje koji se odnosi na vezni dio terena, kako je već ranije objašnjeno broji parove ulaznih vektora u kojima lopta napreduje prema protivničkom golu. Omjer tog broja i ukupnog broja parova je koeficijent kojim se vrši korekcija do sada donesenih odluka. Ako je on veći od 0.5 stare se odluke pojačavaju, inače se oslabljuju.

Skup kojim se uči napadački dio strategije ocjenjuje se po dva kriterija: provedenom vremenu oko protivničkog gola u odnosu na ukupno trajanje utakmice i eventualnim zgoditkom. Ako prijateljski tim postigne gol, sekvenca odluka koja je dovela do gola posebno se pojačava i stavlja na učenje. Ako ne dođe do gola, koeficijent učenja se formira omjerom vremena.

U treniranje po svim skupovima se uvodi faktor težine. Faktor težine povećava ili smanjuje kriterije po kojemu se sekvenca odluka smatra uspješnom ili neuspješnom. Tako npr. prvi skup ima kriterij od 1/3 ukupnog trajanja utakmice, koji se koeficijentom povećava ili smanjuje. Ovaj faktor uveden je za potrebe kasnijeg treniranja kada se ovi osnovni uvjeti redovito ispune, pa sustav prestane napredovati.

Nakon kreiranja skupova, neuronska mreža se trenira standardnom metodom adaptivnog učenja s povratnim prostiranjem izlazne pogreške.



Slika 12. Grafovi učenja. Na gornjem grafu je moguće vidjeti tri krivulje ukupne kvadratne pogreške (sum squared error) po svakom skupu za učenje. Na donjem grafu možemo vidjeti faktor učenja (learning rate) i njegovu promjenu.

Adaptivno učenje s povratnom propagacijom pogreške je metoda odabrana zbog prilagodbe faktora učenja. S obzirom da funkcija odlučivanja nije poznata, ukupna kvadratna pogreška će biti pozitivna i konvergirati prema nekoj konstantnoj vrijednosti.

5.8. Neuronska mreža za predviđanje kretanja loptice

Predviđanje kretanja loptice konstruira se na znatno drugačiji način.

Da bi predviđanje bilo uspješno, potrebno je u ulaznom vektoru prezentirati podatke o prijašnjim stanjima na ploči i pozicijama loptice.

Ulazni vektor neuronske mreže sastoji se od 15 posljednjih vektora stanja s igrače ploče, komprimiranih u 3 vektora. To znači da srednja vrijednost 5 vektora čini jedan komprimirani vektor.

Tri komprimirana vektora, spojena zajedno daju ulazni vektor neuronske mreže. Taj vektor je dimenzionalnosti 3*29=87.

Izlaz neuronske mreže čini dvodimenzionalni vektor koji interpretiran daje predviđene koordinate loptice za pet vremenskih koraka.

Neuronska je mreža izgrađena od 3 sloja, od kojih prva dva imaju po 180 neurona, a treći ima 2 neurona.

5.9. Treniranje mreže za predviđanje kretanja loptice

Skup za treniranje izgrađuje se komprimiranjem i spajanjem ulaznih vektora, te uparivanjem s budućim pozicijama loptice.

Mreža se trenira adaptivnim algoritmom propagacije pogreške.

6. Eksperimenti i rezultati

6.1. Problem određivanja parametara neuronskih mreža, razvojne okoline i simulatora

Intuitivno je razumljivo da će korištenje dviju različitih neuronskih mreža davati različite rezultate na performanse sustava. Već ranije spomenuti faktori kojima se korigira i usmjerava učenje također bitno utječu na brzinu poboljšanja, a ponekad i pogoršanja performansi.

Još uvijek ne postoji model po kojem bi se unaprijed mogla određivati optimalna topologija neuronske mreže za zadani problem. Tako da je dizajniranje neuronskih mreža često prepušteno programerovoj intuiciji i iskustvu, a može biti vođeno eksperimentalnim rezultatima.

Autor ovog rada se problemom određivanja optimalne topologije neuronskih mreža za zadani problem bavio u sklopu rada na znanstvenom članku «Optical Character Recognition of Seven-segment Display Digits Using Neural Networks»[21], tako da su arhitektura i oblik neuronskih mreža rezultat njegova iskustva. Jedan od rezultata pokazuje da se za «dovoljno dobru» klasifikaciju n-dimenzionalnog uzorka u m klasa, općenito neuronska mreža s 3 sloja dobro ponaša. Jedna od zanimljivih karakteristika takvih mreža je i da prva dva sloja obično imaju nešto malo više od n neurona. Vodeći se tim opažanjima dizajnirane su i neuronske mreže korištene unutar ovog projekta.

Jedan od problema s kojim se susreće znanost o inteligentnim sustavima je problem prenaučenosti[22]. Skup uzoraka za učenje može u sebi sadržavati šum, pa i krivo klasificirane uzorke koje će onda neuronska mreža krivo naučiti. Međutim svojstvo prenaučenosti najviše neutralizira svojstvo generalizacije.

[21] Za potrebe tog rada je razvijen sustav koji za skup uzoraka i klasifikacija, genetskim algoritmima nastoji naći optimalnu topologiju mreže.
[22] Eng. overfitting

Svojstvo generalizacije neuronskih mreža je jako bitno svojstvo za ovaj projekt, s obzirom na narav načina treniranja i napredovanja performansi. Njime se postiže da neuronska mreža za slične situacije reagira na isti ili vrlo sličan način.

Pretjerano velikim brojem neurona u slojevima se drastično proširuje aproksimacijska mogućnost neuronske mreže. Samim time se i povećava šansa dolaska u stanje prenaučenosti.

Premalenim brojem neurona aproksimacija funkcije odlučivanja je slaba, i neuronska mreže u sebe može spremiti jako malo «znanja».

Balansirani broj neurona je tako presudan za uspješnost sustava za upravljanje strategijom.

Drugi aspekt predstavljaju parametri okoline. Ako pred sustav koji uči stavimo prelagani zadatak, on će ga riješiti i napredovanje će prestati. Ako pred sustav stavimo pretežak zadatak, onda sustav ne može naći rješenje i do njega vjerojatno nikada neće doći.

Ovaj problem riješen je uvođenjem faktora težine. On se postupno povećava kako sustav napreduje, tako da je pred sustavom za učenje konstantan izazov.

Parametri simulatora se uglavnom tiču simuliranja protoka vremena i fizike igrališta i robota. U okviru ovog rada koriste se parametri i konfiguracijske datoteke iz prethodnih radova studenata na simulatoru.

6.2. Moduli za testiranje

Unutar simulatora postoje dva modula sa kojima se može vršiti testiranje. Modul 1[23], je implementacija strategije koja je orijentirana na obranu. Roboti rijetko prelaze u napad i nemaju razrađen sustav koordinacije. Modul 2[24] orijentiran je na napad, uz nešto slabije obrambene karakteristike.

[23] Interno nazvan CoreanStrategy
[24] Interno nazvan NabrijaniCoreanStrategy

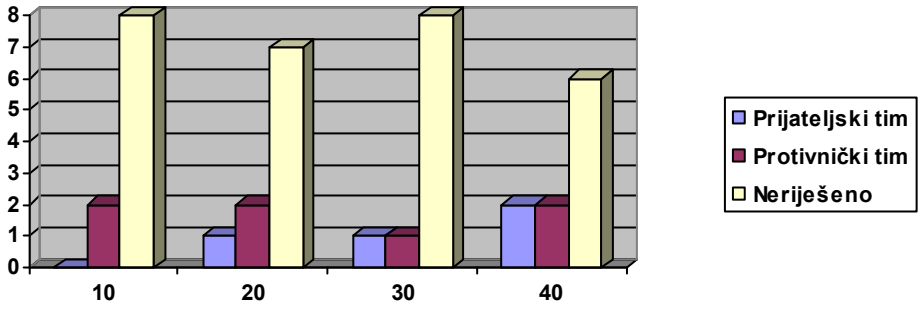
Modul u kojem je realizirana strategija bazirana na neuronskim mrežama zove se MatlabModul i on je suprotstavljen Modulu 1 i 2 u tijekom testiranja.

6.3. Rezultati treniranja sustava protiv Modula 1

Neuronska mreža se inicijalizira. Matlab inicijalizaciju provodi tako da sve vrijednosti vektora težina i pragova postavi na neki slučajni broj.

Rezultat će se promatrati kroz ukupan broj golova koji realizira svaki od timova u deset utakmica. Vrijeme isteka utakmice je 5000 vremenskih koraka i utakmice koje u tom vremenu ne završe golom se smatraju neriješenima.

Nakon svake utakmice slijedi treniranje.



Slika 13. Rezultat testiranja na Modulu 1. Stupci prikazuju broj golova određenog tima po utakmici.

U početku se roboti ponašaju prilično kaotično i neorganizirano, međutim već nakon nekoliko utakmica se mogu početi mjeriti sa protivničkim timom.

Broj utakmica kojima je isteklo vrijeme bez gola je razmjerno velik, što je i očekivano s obzirom na obrambeni stil igre Modula 1. Također, oba modula koriste iste implementacije uloga što ih čini ravnopravnim protivnicima.

Promatrajući rezultate protivničkog tima može se primijetiti kako je njihova kvaliteta igre konstantna kroz svih 40 utakmica.

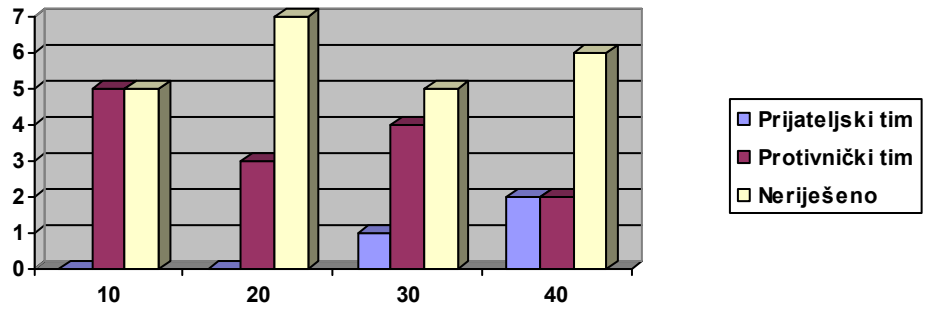
Međutim napredak u kvaliteti igre je vidljiv kod sustava upravljanog neuronskom mrežom. Postupni porast broja golova posljedica je sakupljanja znanja i pretraživanja skupa mogućih rješenja.

Promatrajući samu igru može se primijetiti da je veći broj golova posljedica sve agresivnije igre, i sve češćeg prebacivanja loptice u prostor protivničkog gola. Srednja vrijednost x-koordinate[25] loptice u 40. utakmici je 1.7, što znači da se statistički veći dio utakmice igra na protivničkom terenu.

Nameće se zaključak da će nakon dovoljnog broja utakmica omjer uspješnosti Matlab Modula naspram Modula 1 konvergirati u neku konstantu vrijednost.

6.4. Rezultati treniranja sustava protiv Modula 2

Isti uvjeti kao i za eksperiment sa Modulom 1 se primjenjuju na Modul 2.



Slika 14. Rezultat testiranja na Modulu 1. Stupci prikazuju broj golova određenog tima po utakmici.

Modul 2 se po ponašanju razlikuje od Modula 1 po agresivnijem pristupu igri. To je vidljivo na početku igranja kada Modul 2 potpuno dominira u igri. Tijekom prvih 20 utakmica MatlabModul uspije razviti nešto učinkovitiju obranu, tako da u trećoj seriji utakmica pomalo razvija napadački stil.

U četvrtoj seriji utakmica je već moguće primijetiti smanjenu dominaciju Modula 2, i mogućnost ravnopravne igre. Glavni uzrok takvog rezultata je u pojačanju

[25] x-koordinata može biti vrijednost iz intervala [0, 2.2]

agresivnosti u igri, tako da se težište sve češće prebacuje na protivnički dio terena. Modul nakon 40 utakmica balansira između napadačke i obrambene taktike.

6.5. Analiza rezultata

Promatrajući rezultate u cjelini vidljiv je napredak u poboljšanju performansi sustava. Sustav iz početnog kaotičnog ponašanja, a zbog čestih primljenih golova prvo razvija obranu. Kada obrana postane dovoljno dobra da loptica sve manje boravi u prvoj trećini igrališta sve češće vezni dio dolazi do izražaja. Napadački stil igre se razvija najkasnije.

Tijekom razvoja strategije mogu se primijetiti periodi u kojima sustav napreduje, ali i oni u kojima performanse opadaju. To je rezultat pretraživanja prostora mogućih rješenja slučajnim pomacima. Međutim, unatoč periodima opadanja performansi sustav ukupno gledano povećava kvalitetu igre.

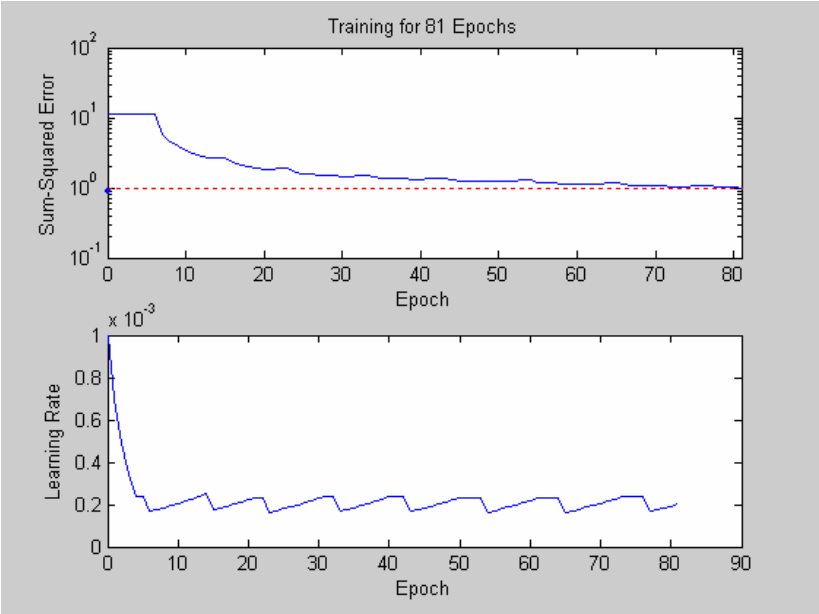
Zanimljivi rezultati dobivaju se ako neuronsku mrežu naučenu na Modulu 1 koristimo na Modulu 2 i obratno. Općenito gledano u oba slučaja, ukupni rezultat je nešto lošiji nego protiv modula na kojem se neuronska mreža učila. Promatrajući to iz perspektive neuronske mreže kao aproksimacijske funkcije mogli bismo reći da neuronska mreža uči kako pobijediti taj specifičan modul protiv kojega igra.

Najbolji rezultati dobivaju se kada se treniranje vrši na raznim modulima naizmjence.

Zanimljivo je primijetiti da kada jednom sustav nauči dobro igrati protiv nekog modula, te se treniranje uglavnom obavlja za napadački dio igre, obrana s vremenom postaje sve lošija. Takvo «zaboravljanje» obrane je posljedica korekcije težinskih faktora unutar neuronske mreže tj. slabijom aproksimacijom dijela funkcije koji dugo nije bio u skupu za učenje.

6.6. Rezultati treniranja neuronske mreže za predviđanje kretanja loptice

Poboljšanje performansi za predviđanje kretanja loptice je nešto lakše izmjeriti. Potrebno je samo pratiti ukupnu srednju kvadratnu pogrešku prilikom treniranja.



Slika 15. Graf ukupne kvadratne pogreške za tijekom treniranja neuronske mreže za predviđanje kretanja loptice

Iz slike 15. je vidljivo kako se neuronska mreža već nakon nekoliko desetaka epoha istrenira tako da srednje kvadratna pogreška zadovolji cilj[26]. U sljedećoj sličnoj igri neuronska mreža će vrlo dobro predvidjeti ponašanje loptice.

Situacije za koje neuronska mreža nije nikad dobila primjer za učenje, ne predviđaju se toliko dobro, kao one za koje je dobila primjer za učenje. Obzirom da se radi o velikoj mreži sa mnogo neurona i mnogo veza, neuronska mreža predstavlja vrlo složenu aproksimacijsku funkciju, što znači da u sebi može pohraniti mnogo znanja. Što se više utakmica odigra, te što se više različitih podataka koristi za treniranje mreže, to je predviđanje kretanja loptice učinkovitije.

[26] Eng. Goal – vrijednost ukupne kvadratne pogreške pri kojoj se treniranje prekida

7. Zaključak

Inteligentni sustavi sa sposobnošću samostalnog učenja tema su mnogih znanstvenih istraživanja. Potencijalna korist jednog takvog sustava je velika, i teško je zamisliti granu ljudske djelatnosti u kojoj se takav sustav ne bi mogao primijeniti.

Motivacija za istraživanje mogućnosti primjene neuronskih mreža može se naći u činjenici da je tijekom povijesti oponašanje sustava iz prirode dovelo do mnogih velikih otkrića. Neuronska mreža pokazuje se kao vrlo dobar model za aproksimacije funkcije najviše zbog nelinearnih svojstava i jednostavnog treniranja. Uzimajući u obzir i sposobnost generalizacije, neuronska mreža može poslužiti kao komponenta inteligentnog sustava koja objedinjuje otkrivanje i pohranu novog znanja.

U okviru ovog rada predložen je model u kojem se neuronska mreža koristi kao osnovni dio inteligentnog sustava, te integrirani model sakupljanja znanja. Također je pokazana i mogućnost primjene neuronske mreže na pomoć pri automatskom upravljanju predviđanjem varijabli sustava.

Osnovna prednost ovog sustava je samostalnost. Uz dovoljno danog vremena sustav će postupno napredovati bez uplitanja čovjeka. Druga prednost samostalnosti je nesubjektivno otkrivanje znanja. Ono dovodi do znanja koje ljudski programer zbog oslanjanja na intuiciju i iskustvo možda nikada ne bi implementirao.

Glavni nedostatak proizlazi iz istog razloga kao i osnovna prednost a to je neselektivno pretraživanje prostora mogućih rješenja. Postupak je računski iznimno zahtjevan, a ne postoji garancija da sustav neće konvergirati u lokalni optimum. Zbog naravi načina na koji se traži rješenje, performanse sustava često imaju tendenciju pogoršanja, i teško je predvidjeti kado će se ponašati. Također, sustav ima tendenciju da se prilagodi trenutnom protivniku, i da se nauči protiv njega igrati dovoljno dobro, što se negativno odražava na generalne performanse protiv različitih protivnika.

Pa ipak, prvi rezultati su ohrabrujući, a treniranje tijekom izrade projekta je rezultiralo nekolicinom izvrsnih konfiguracija.

Ovako modeliran inteligentni sustav se uz minimalne promjene može primijeniti na bilo koji problem odlučivanja koji se može svesti na dodjelu uloga.

8. Literatura

[1] *Artificial Intelligence*, Wikipedija članak, 2009, URL: http://en.wikipedia.org/wiki/Artificial_intelligence

[2] Marek Obitko: *Prediction using neural networks*, 1999, URL: http://www.obitko.com/tutorials/neural-network-prediction/conclusion.html

[3] Prof. Dr. Darko Tipurić, Predavanje: *Vrste odluka, stilovi odlučivanja i pristupi odlučivanju*, 2009, URL: http://web.efzg.hr/dok//OIM/dtipuric//2-Vrste odluka, stilovi i pristupi odlučivanju -2009.pdf

9. Sažetak

Ovaj rad prezentira model inteligentnog sustava temeljenog na neuronskim mrežama koji upravlja višerobotskim sustavom dodjelom robotskih uloga. Pretragom prostora svih mogućih odluka, postupno se postižu sve bolje performanse sustava. Neuronske mreže se također koriste i kao nadopuna automatskom upravljanju predviđanjem varijabli sustava, kao što je pozicija loptice na igralištu.

Sustav se trenira i testira na dva ranije izgrađena modula, te prvi rezultati pokazuju poboljšanje performansi nakon više odigranih utakmica.

Ključne riječi: strategija, neuronske mreže, višerobotski sustavi..

10. Abstract

This work present a model of intelligent system based on neural network, who manages the multi-robot system by assigning robot roles. By searching the space of all possible decisions, system performance is increasing. Neural networks are also used as enhancement of automated control, by predicting system variables, such as location of ball in playground.

System is trained and tested on two formerly built modules, and first results show improvement after several played matches.

Keywords: strategy, neural networks, multirobot systems.

11. Životopis

Rođen sam 27.11.1984. godine u Rijeci. Ondje sam završio osnovnu školu, a zatim prirodoslovno-matematičku gimnaziju (Gimnazija Andrije Mohorovičića u Rijeci). Maturirao sam 2003 godine i iste sam godine upisao Fakultet elektrotehnike i računarstva u Zagrebu, smjer računarstvo.