Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Erik Rakušček

Proceduralna animacija skupinskega lova volkov

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Iztok Lebar Bajec
Somentor: doc. dr. Jure Demšar

Ljubljana, 2018

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Proceduralna animacija skupinskega lova volkov

Tematika naloge:

Nadgradite diplomsko delo Mitje Goriška (Krdelo volkov: modeliranje in simulacija skupinskega plenjenja, FRI UNI-LJ, 2017) do te mere, da bo slednji za namene računalniške igre lahko prikazal vizualno prepričljivo dogajanje plenjenja volkov. Rezultate primerjajte s prosto dostopnimi videoposnetki s spleta in jih kritično komentirajte.

Rad bi se zahvalil mentorju izr. prof. dr. Iztoku Lebarju Bajcu in somentorju doc. dr. Juretu Demšarju za hitro odzivnost, pomoč in nasvete pri iskanju literature ter za dodatno gradivo, ki mi je bilo v veliko pomoč pri dokončanju diplomske naloge.

Posebej bi se rad zahvalil svoji družini in prijateljem za podporo med izdelavo diplomskega dela.

Tatiju Dominiku.

Kazalo

Povzetek

Naslov: Proceduralna animacija skupinskega lova volkov

Avtor: Erik Rakušček

Računalniške simulacije postajajo vse pogostejši način izobraževanja in zabave, saj nam omogočajo opazovanje naravnih fenomenov, ki bi jih zelo težko videli v živo. Eden izmed takšnih fenomenov je tudi skupinsko gibanje in plenjenje volkov (*Canis lupus*). Čeprav obstaja veliko video posnetkov na to temo, pa nam le-ti ne omogočajo interaktivnega okolja, kjer bi lahko poljubno spreminjali pogoje in opazovali razlike. Za popolno opazovanje fenomena pod različnimi pogoji torej potrebujemo računalniško simulacijo.

Simulacijo lahko razdelimo v dva dela. V prvem delu skušamo prikazati preiskovanje teritorija volkov in sprehajanje plena. Za osnovo vzamemo Boids algoritem iz vira (Reynolds (1987), Flocks, herds and schools: A distributed behavioral model, DOI: 10.1145/37402.37406) in ga nadgradimo z dodatnimi težnjami posameznikov v simulaciji. V drugem delu se osredotočimo na plenjenje. Na podlagi modela iz vira (Gorišek (2017), Krdelo volkov: modeliranje in simulacija skupinskega plenjenja, FRI UNI-LJ) implementiramo obkoljevanje in napad volkov ter obrambo plena. Model prilagodimo in nadgradimo za bolj realističen prikaz vedenja.

Na podlagi testiranja in primerjave simulacije s prosto dostopnim video gradivom na spletu, smo ugotovili, da se je popolni realnosti skoraj nemogoče približati. Kljub temu se nam je uspelo dogajanju v naravi približati dovolj dobro za uporabo modela v zabavni industriji ali v osnovnih izobraževalnih programih.

Ključne besede: simulacija, lov, sprehajanje, obkoljevanje, umetno življenje.

Abstract

Title: Procedural animation of wolf-pack hunting

Author: Erik Rakušček

Computer simulations are increasing in popularity as a source of education and entertainment. They allow us to observe and study natural phenomena, we could hardly ever witness in real life. One such phenomenon is group hunt of a wolf-pack (*Canis lupus*). Even though there are plenty of videos on the topic, they do not enable us to interact with the environment and change the conditions at our will. That is why a computer simulation can help us study the phenomenon.

Our simulation can be divided into two sections. Our goal in the first section is to simulate wolf-pack's search for prey and the prey's wandering. We create a model based on Boids (Reynolds (1987), Flocks, herds and schools: A distributed behavioral model, DOI: 10.1145/37402.37406) and upgrade it with additional tendencies of the agents. In the second section of the project, we focus on the hunt. We implement surrounding and attacking behaviour for the wolves and defensive behaviour for the prey, based on a model from (Gorišek (2017), Krdelo volkov: modeliranje in simulacija skupinskega plenjenja, FRI UNI-LJ). We adjust and upgrade said model to display a more realistic behaviour.

Based on our testing and comparison of our model with the freely available videos on the internet, we found out that it is quite difficult to simulate wolf hunt the way it happens in reality. Nevertheless, we managed to create a

realistic enough model that could be used in the entertainment industry or basic educational programs.

Keywords: simulation, hunt, wandering, surrounding, artificial life.

Poglavje 1

Uvod

Število računalniških simulacij iz dneva v dan narašča, saj so najboljši način, kako ljudem približati neko tematiko. Uporabljajo se tako v zabavne, kot tudi znanstvene namene. Prav zaradi tega smo se odločili simulirati plenjenje skupine volkov in s tem širši javnosti predstaviti ta naravni fenomen.

Volkovi so zmožni zelo hitro presoditi razmerje med ceno in koristjo napada na določen plen. Največkrat plenijo živali, ki so zaradi starosti ali naravnih okoliščin oslabljene [1]. Velikokrat najprej izmučijo plen in ga šele nato napadejo. Tako zmanjšajo verjetnost poškodb, četudi zaradi tega plenjenje lahko traja do 11 ur. Skozi evolucijo so se volkovi naučili, da imajo največje možnosti za uspešen lov, če se združijo v krdela. Krdela navadno štejejo okoli 7 volkov in lahko od enega uspešnega lova na večjo žival jedo več dni. Najbolj fascinantno pa je dejstvo, da so se volkovi med napadom sposobni organizirano premikati in obkoliti plen.

V prvem delu diplomske naloge prikažemo preiskovanje teritorija volkov, plenu pa dodamo logiko naključnega sprehajanja. Zgledujemo se po algoritmu Boids [2]. V drugem delu simuliramo plenjenje na podlagi modela, ki ga je razvil Gorišek [3]. Volkovi si za tarčo primarno izbirajo mladiče, saj le-ti niso sposobni samoobrambe. Med lovom se volkovi skušajo izogibati nevarnim proti-napadom odraslih, močnejših živali. Če volkovi mladičev ali oslabljenih odraslih živali ne najdejo, lahko zaradi grožnje lakote napadejo

tudi močnejši odrasel plen [4].

Z našo simulacijo se želimo približati stanju v naravi. Ker pa tematika plenjenja volkov še ni povsem znanstveno raziskana in v naravi obstaja ogromno različnih pogojev, ki se stalno spreminjajo, lahko ustvarimo le približek obnašanja volkov v realnosti. Kljub temu pa uspemo ustvariti model, ki dovolj dobro prikazuje obnašanje volkov za neznanstvene namene. Na primer, za uporabo v računalniških igrah ali osnovnih izobraževalnih programih.

Poglavje 2

Uporabljena orodja in tehnologije

2.1 Unity

Unity je igralni pogon, ki so ga razvili pri Unity Technologies ApS leta 2005. Od takrat naprej je v konstantem razvoju, z rednimi nadgranjami vsakih nekaj mesecev. Unity omogoča kreiranje iger in simulacij na zelo intuitiven način. V njem je mogoče ustvariti tudi kompleksne simulacije že z relativno malo pisanja programske kode, zato je ta igralni pogon primeren tako za začetnike, kot tudi izkušene programerje. Unity podpira programska jezika C# in JavaScript. Nudi tudi izvoz projektov na veliko različnih platform in omogoča enostaven prenos projektov med platformami. Prav zaradi tega je postal zelo popularen med razvijalci. Obstajajo tri verzije Unity-ja: personal, plus in pro [5].

V sklopu diplomske naloge uporabljamo Unity personal edition verzijo 2017.3.1f1. Za Unity smo se odločili zato, ker smo se želeli posvetiti predvsem zalednemu delu simulacije, poleg tega pa na hiter način model tudi vizualizirati.

2.2 Visual Studio 2017

Visual Studio 2017 je razvijalsko okolje, ki so ga razvili pri Microsoft-u. Uporablja se za razvoj računalniških programov, spletnih strani, aplikacij... Tudi Visual Studio je na voljo v treh različnih verzijah: Community, Professional in Enterprise.

V diplomski nalogi uporabljamo Community verzijo Visual Studia. Vsa programska koda modela je napisana v Visual Studio predvsem zato, ker je zelo dobro integriran v Unity. Poleg tega nam omogoča boljši pregled nad kodo in razhroščevanje kot Unity-jev prvotni urejevalnik kode MonoDevelop.

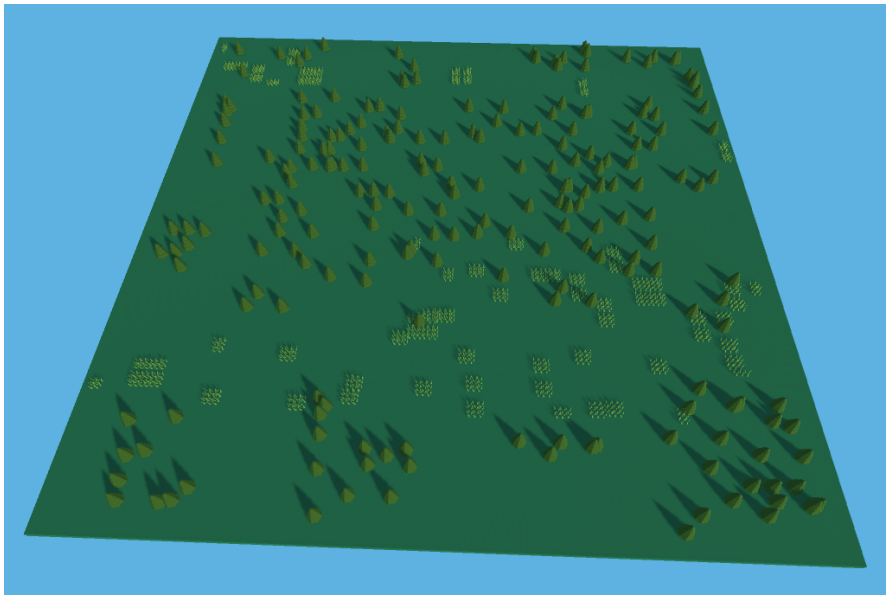
2.3 C#

Vso programsko kodo v okviru diplomske naloge smo napisali v jeziku C#. C# je objektno orientiran programski jezik, ki so ga razvili pri Microsoft-u v okviru razvoja ogrodja .NET. Pri skladnji se zgleduje po številnih drugih programskih jezikih, najbolj izrazito pa po C, C++ in Javi. Ker sta nam programska jezika C in Java bližje kot Javascript in imamo z njima več izkušenj, smo se odločili za uporabo jezika C#.

Poglavje 3

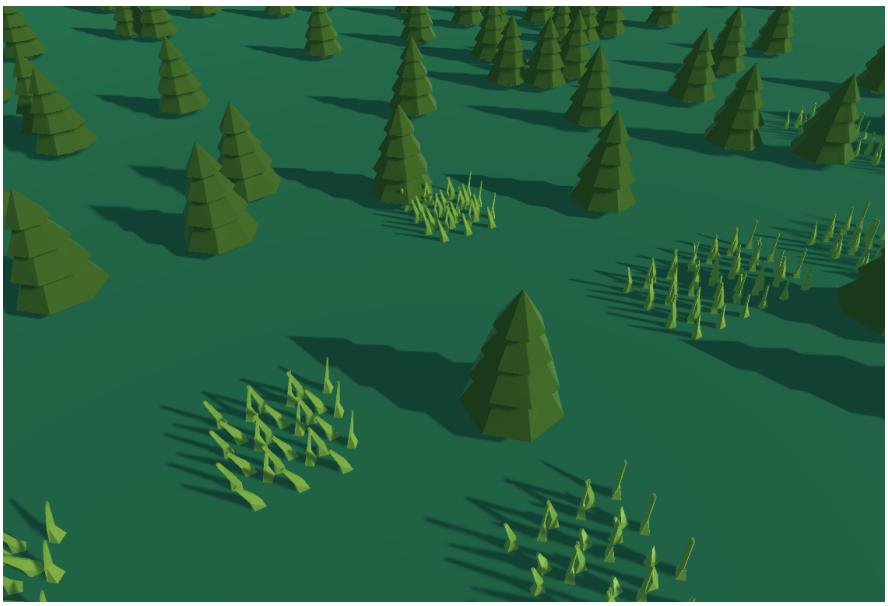
Okolje simulacije

Volkovi so teritorialne živali [4], zato celotno okolje simulacije pripada eni družini volkov. Velikost teritorija družin volkov ni enotna, saj je odvisna od več faktorjev, kot so velikost družine volkov, gostota plena, starost mladičev [6]... Zaradi potrebe po vzdrževanju teritorija, navadno pride do tega, da si volkovi ustvarijo dovolj velik teritorij za svoje potrebe, obenem pa poskrbijo da je, zaradi cene vzdrževanja, čim manjši [7].



Slika 3.1: Okolje simulacije izdelano v Unity.

V realnosti lahko velikost teritorija ene skupine volkov sega od 30 km^2 pa vse do 6000 km^2. V našem modelu je velikost teritorija mogoče poljubno definirati s pomočjo parametrov. Ker pa je cilj diplomske naloge izvesti simulacijo plenjenja volkov v nekem določenem času, velikost teritorija priredimo tako, da simulacija ne traja več kot 5 minut. Zato smo nastavili privzeto velikost teritorija na približno 55 km^2 (slika 3.1).



Slika 3.2: Naravne ovire (drevesa in trava) v Unity.

Za dosego višje stopnje realizma smo v okolje dodali tudi drevesa in travo (slika 3.2). Drevesa predstavljajo neprehodno oviro, visoka trava pa težje prehodno oviro. Tako plen kot volkovi se pri gibanju skozi okolje izmikajo drevesom. V primeru, da plen oziroma volk zabrede v visoko travo pa ga le-ta upočasni za določen delež njegove normalne hitrosti.

Naravne ovire vplivajo tudi na vidno polje plena. V primeru, da se volk skrije za drevo, plen čez določen čas izgubi sled za njim. Kljub temu pa volk, zaradi poznavanja teritorija, še vedno pozna lokacijo plena, zato s tem manevrom pridobi prednost pri napadu. Plen se namreč ne bo branil pred njim, dokler ga spet na zagleda ali zasliši. Volk se lahko skrije tudi v visoko travo. Tudi v tem primeru plen čez določen čas izgubi sled za volkom. Čeprav

plenilca visoka trava upočasni, mu le-ta daje tudi prednost, saj se tako lahko neopažen približa plenu na krajšo razdaljo.

Ker so naravne ovire za plen bolj nevarne kot odprti teritoriji, se bo plen skušal izogibati oviram. Iskal bo kar se da odprto ozemlje, kjer ima največ možnosti za zgodnjo detekcijo plenilca in uspešno obrambo pred njim. Najnevarnejše za plen je, da zabrede v visoko travo, saj si s tem močno zmanjša možnosti za detekcijo plenilca. Zato se plen izogiba tudi visoki travi.

Poglavje 4

Vedenje volkov

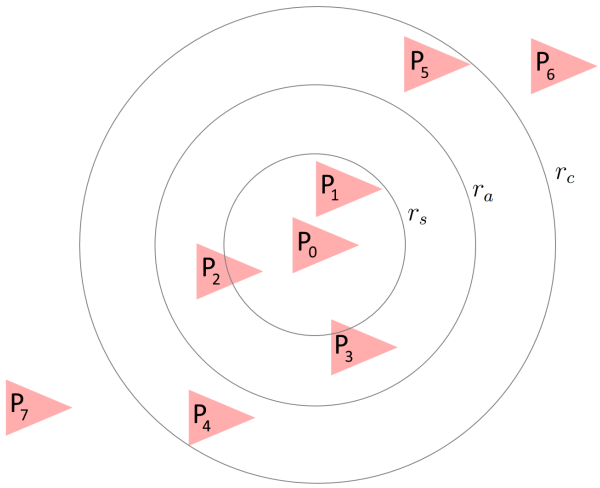
V naravi obstajajo razlike v načinu obnašanja volkov v divjini in volkov v ujetništvu [8]. Naš model plenjenja volkov je osredotočen na obnašanje volkov v divjini. Ker so volkovi znani po tem, da plenijo v skupinah, smo pri implementaciji logike volkov stremeli k temu, da se volkovi, tako kot v naravi, povezujejo v krdela. Ko volkovi odkrijejo plen in ga izberejo za tarčo napada, pričnejo z obkoljevanjem. Tekom obkoljevanja, ko volk ugotovi, da je varno napasti plen, izvede napad. Uspešnost plenilčevega napada je odvisna od obrambe plena in splošnega trenutnega stanja v okolju kjer napad poteka. Vedenje volkov, v sklopu lova na plen, lahko torej razdelimo na 3 faze: iskanje plena, obkoljevanje plena in napad na plen.

4.1 Iskanje plena

Iskanje plena temelji tako na naključnosti kot na logičnem sklepanju volkov. Vedenje volkov smo implementirali na podlagi računalniškega modela za simulacijo letenja ptic v jati Boids [2, 9], ki ga je razvil Craig Reynolds.

Ta model je zasnovan na ravni posameznika, pri katerem vsak izmed simuliranih agentov skuša zadovoljiti tri težnje: kohezijo (angl. *cohesion*), poravnavo (angl. *alignment*) in razmik (angl. *separation*). Rezultat vsake težnje je sila, ki poskrbi za premik posameznika.

Stanje sveta okoli posameznika je definirano skozi njegove sosede, ki se nahajajo znotraj določenega radija, ki določa območje zaznave.



Slika 4.1: Prikaz parametrov zaznavanja pri modelu Boids.

Zunanji radij r_c določa območje zaznavanja drugih plenilcev. Med r_a in r_c so posamezniki, ki so predaleč, zato se jim skušamo približati s kohezijo. Med r_a in r_s so posamezniki na ravno pravi razdalji, zato skuša opazovani volk s poravnavo z njimi uskladiti smer. Posamezniki, ki so bližje od r_s so opazovanemu volku preblizu, zato se skuša z razmikom od njih oddaljiti. Volkovi, ki se nahajajo dlje od r_c, na opazovanega volka ne vplivajo.

V primeru na sliki 4.1, na težnjo razmika opazovanega plenilca P_0 vpliva plenilec P_1, na težnjo poravnave plenilca P_2 in P_3, na težnjo kohezije pa plenilca P_4 ter P_5. Na opazovanega volka nimata vpliva plenilca P_6 in P_7.

Končni premik posameznika $\vec{v}_i$ izračunamo tako, da sile teženj pomnožimo s posameznimi utežmi in jih seštejemo:

$$\vec{v}_i = \vec{f}_{c,i} \cdot w_{c,i} + \vec{f}_{a,i} \cdot w_{a,i} + \vec{f}_{s,i} \cdot w_{s,i}. \tag{4.1}$$

Premik posameznika izvedemo tako, da posameznikovi trenutni lokaciji prištejemo vektor premika $\vec{v}_i$.

S težnjo kohezije se opazovani posameznik skuša približati ostalim volkovom, ki jih vidi in so od njega preveč oddaljeni (dlje od r_a ampak bližje od r_c):

$$\vec{f}_{c,i} = \frac{\sum_{j \in N_{c,i}} \vec{p}_j}{|N_{c,i}|} - \vec{p}_i, \tag{4.2}$$

kjer $\vec{f}_{c,i}$ predstavlja silo kohezije, $N_{c,i}$ množico posameznikov, ki so znotraj predpisanih radijev, $\vec{p}_i$ pozicijo opazovanega posameznika, $\vec{p}_j$ pa pozicijo ostalih posameznikov. S tem dosežemo približevanje posameznikov, ki so en od drugega preveč oddaljeni in tako tvorjenje skupin.

S težnjo poravnave, skuša opazovani posameznik poravnati smer in hitrost premika z ostalimi volkovi, ki so dlje od r_s in bližje od r_a:

$$\vec{f}_{a,i} = \frac{\sum_{j \in N_{a,i}} \vec{v}_j}{|N_{a,i}|}, \tag{4.3}$$

kjer $\vec{f}_{a,i}$ predstavlja silo poravnave, $N_{a,i}$ pa množico posameznikov, ki so znotraj predpisanih radijev. S tem dosežemo gibanje posameznikov v isto smer kot organizirane skupine.

S težnjo razmika preprečujemo trke med posamezniki, saj se opazovani posameznik skuša oddaljiti od ostalih volkov, ki so bližje od r_s:

$$\vec{f}_{s,i} = \sum_{j \in N_{s,i}} \frac{\vec{o}_{ij}}{|\vec{o}_{ij}|} \cdot (r_s^2 - |\vec{o}_{ij}|^2), \tag{4.4}$$

kjer $\vec{f}_{s,i}$ predstavlja silo razmika, $N_{s,i}$ množica vseh sosedov opazovanega posameznika, ki so znotraj radija razmika in $\vec{o}_{ij}$:

$$\vec{o}_{ij} = \vec{p}_i - \vec{p}_j. \tag{4.5}$$

Ker je algoritem Boids v osnovi namenjen gibanju ptic v jati, brez prisotnosti plenilcev, smo morali Boids algoritem nadgraditi, da bi se približali naravnemu premikanju posameznikov in povzročili preiskovanje teritorija. Model smo nadgradili tako, da smo dodali še gibanje skupine proti nekemu

cilju, logiko za naključno sprehajanje, zadrževanje znotraj območja ter logiko za izmikanje oviram. Sile dodatnih teženj najprej normaliziramo, nato pa pomnožimo s posameznimi utežmi in jih prištejemo h končnemu premiku:

$$\begin{aligned}\vec{v_i} = \vec{f}_{c,i} \cdot w_{c,i} + \vec{f}_{a,i} \cdot w_{a,i} + \vec{f}_{s,i} \cdot w_{s,i} + \vec{f}_{gt,i} \cdot w_{gt,i} + \\ \vec{f}_{w,i} \cdot w_{w,i} + \vec{f}_{e,i} \cdot w_{e,i} + \vec{f}_{oa,i} \cdot w_{oa,i}.\end{aligned} \tag{4.6}$$

Gibanje proti cilju

Medtem ko večina teženj generira sile, ki se stalno spreminjajo, se sila pri težnji za gibanje proti cilju spremeni zgolj občasno - ko volkovi dosežejo trenutni cilj. S to težnjo v fazi iskanja plena dosežemo premikanje volkov po celotnem območju. Težnja za gibanje proti cilju generira silo $\vec{f}_{gt,i}$, ki kaže od plenilca proti končni točki.

Vsak plenilec si na začetku simulacije naključno določi končno točko znotraj teritorija. Ko tekom simulacije plenilec pride dovolj blizu svoje končne točke, naključno določi novo končno točko, tako da kot med plenilčevo trenutno usmerjenostjo in novo končno točko ni večji od 90°. Z omejitvijo kota nove končne točke preprečimo, da bi pri plenilcu prišlo do nenadnega obračanja za več kot 90°. Tako pot plenilca v našem algoritmu izgleda bolj enotna in menjava končne točke postane skoraj neopazna.

Plenilec spremeni svojo končno točko tudi ko zagleda plen ali zazna njegov vonj. Plen namreč izpušča vonj vsakih nekaj sekund. Vonj je predstavljen kot objekt z lokacijo in močjo. Volk si zapomni lokacije in moč vonjav, ki jih je zaznal. Moč vonjav s starostjo upada. Da bi volk na podlagi vonja lažje izsledil plen, mu ob zaznavi vonja nastavimo končno točko na lokacijo, kjer je zaznal najintenzivnejšo vonjavo.

Dodatno smo nadgradili izračunavanje sile $\vec{f}_{gt,i}$ pri skupini volkov. Pri tem smo izkoristili parameter *cw*, ki ponazarja vpliv posameznega volka na ostale volkove. Skupine volkov imajo namreč hierarhično strukturo (alfa, beta in omega volkovi), v kateri imajo določeni posamezniki večji vpliv na vedenje skupine [10]. Če je v skupini več kot 1 plenilec, poiščemo plenilca z

najvišjo vrednostjo parametra cw, nato pa začnemo premikati končno točko vseh ostali plenilcev proti končni točki plenilca z najvišjim cw. Končne točke premikamo dokler ne postanejo enake končni točki plenilca z najvišjim cw. S tem dosežemo bolj enotno premikanje skupine volkov in tako ustvarimo na videz bolje organizirano krdelo. V programski kodi 4.1 je prikazana implementacija algoritma za izračun sile $\vec{f}_{gt,i}$. Poleg tega v tem delu kode poteka tudi prilagajanje cilja (vrstice 8-22).

Programska koda 4.1: Funkcija za izračun sile $\vec{f}_{gt,i}$ v programu Unity

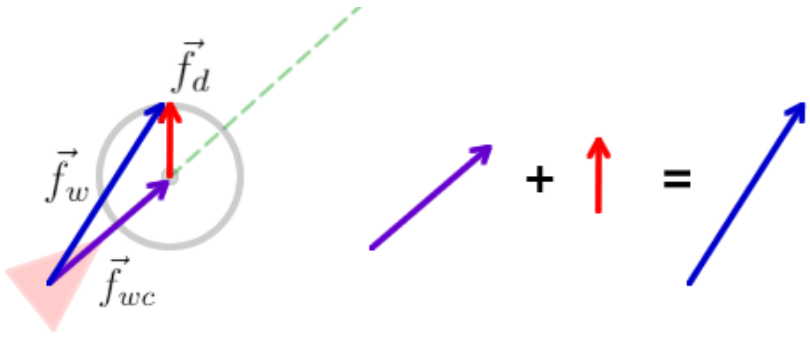
```
1  public Vector3 GlobalTargetForce()
2  {
3      if (distAB(transform.position, globalTarget) < changeGTDist)
4      {
5          globalTarget = RandomTargetPoint();
6      }
7
8      float highestCw = cw;
9      Vector3 changeTo = new Vector3();
10     foreach (GameObject wolf in otherWolves)
11     {
12         if (distAB(transform.position, wolf.transform.position) < rA &&
               wolf.GetComponent<WolfBehaviour>().cw > highestCw)
13         {
14             changeTo = wolf.GetComponent<WolfBehaviour>().globalTarget;
15             highestCw = wolf.GetComponent<WolfBehaviour>().cw;
16         }
17     }
18
19     if (highestCw != cw)
20     {
21         globalTarget += (changeTo - globalTarget).normalized *
               changeGTStep;
22     }
23
24     return (globalTarget - transform.position).normalized;
25 }
```

Funkcija $GlobalTargetForce()$ nam vrne normaliziran vektor v smeri od volka proti njegovemu trenutnemu cilju. Z metodo $distAB(A, B)$ izračunamo razdaljo med točkama A in B, metoda $RandomTargetPoint()$ pa nam vrne naključno točko v teritoriju. Konstanta $changeGTDist$ nam pove na kakšni

razdalji do trenutnega cilja naj volk zamenja svoj cilj, konstanta *changeGTStep* pa nam pove s kakšnim korakom naj volkovi prilagajajo svoj cilj cilju najvplivnejšega volka v krdelu (volk z najvišjim *cw*).

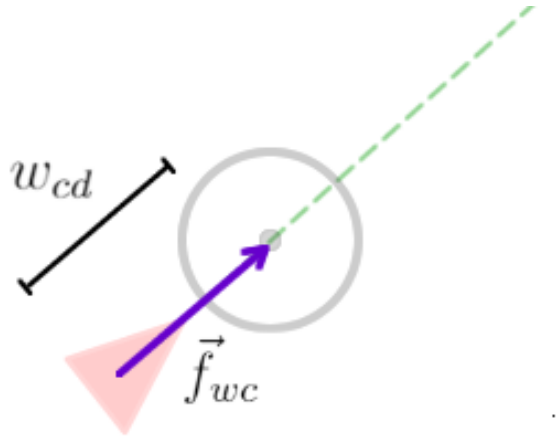
Težnja naključnega sprehajanja

Ker je zgolj s težnjo za gibanje proti končnemu cilju vedenje volkov nenaravno (volkovi se do cilja sprehodijo bolj ali manj v povsem ravni liniji), smo modelu dodali težnjo naključnega sprehajanja. S to težnjo dodamo gibanju volkov premike v rahlo naključni smeri. Rezultat vsega skupaj pa je bolj naravno gibanje. Težnjo smo povzeli po Reynoldsu [11] in Bevilacqui [12].



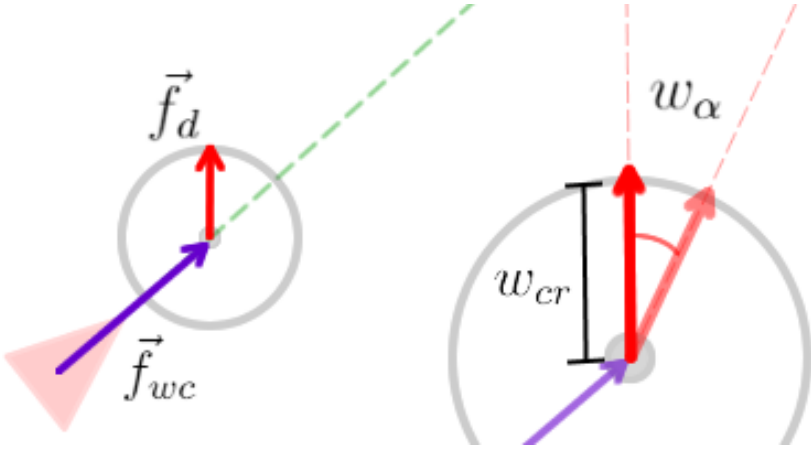
Slika 4.2: Sila težnje naključnega sprehajanja $\vec{f}_{w,i}$ kot vsota sil $\vec{f}_{wc}$ in $\vec{f}_d$ [12].

Silo težnje naključnega sprehajanja $\vec{f}_{w,i}$ dobimo tako, da seštejemo silo kroga naključnega sprehajanja $\vec{f}_{wc}$ in silo odmika $\vec{f}_d$ (slika 4.2). Na slikah (4.2), (4.3), (4.4) je krog naključnega sprehajanja predstavljen s sivo barvo. Pri izračunu sile $\vec{f}_{w,i}$ potrebujemo radij in lokacijo kroga naključnega sprehajanja, medtem ko sam krog služi le lažji predstavi delovanja sile $\vec{f}_{w,i}$.



Slika 4.3: Sila kroga naključnega sprehajanja $\vec{f}_{wc}$ [12].

Silo kroga naključnega sprehajanja $\vec{f}_{wc}$ določimo od trenutne lokacije plenilca v smeri premika, z dolžino w_{cd} (slika 4.3). Parameter w_{cd} je konstanta, ki jo nastavimo pred začetkom simulacije in predstavlja razdaljo od trenutne lokacije plenilca do središča kroga naključnega sprehajanja .



Slika 4.4: Premik sile odmika $\vec{f}_d$ v dveh zaporednih slikah za kot w_α [12].

Silo odmika (angl. *displacement*) $\vec{f}_d$ določimo z začetkom v središču kroga naključnega sprehajanja z dolžino w_{cr}. Smer sile odmika pa se v vsaki sliki simulacije spremeni za w_α stopinj v naključni smeri (slika 4.4). Parametera w_{cr} in w_α sta konstanti, ki ju nastavimo pred začetkom simulacije. Parame-

ter w_{cr} predstavlja radij kroga naključnega sprehajanja, parameter w_α pa kot spremembe smeri sile $\vec{f_d}$ v posamezni sliki. Tako premiku posameznika dodamo nekaj naključnosti, medtem pa preprečimo prevelike spremembe smeri posameznika v zaporednih slikah simulacije.

Zadrževanje znotraj območja

Ker so volkovi teritorialne živali [4], plenijo samo znotraj svojega teritorija. Zato je pomembno, da pri preiskovanju teritorija ne zaidejo izven njegovih meja. To dosežemo s silo robov teritorija $\vec{f}_{e,i}$. Ta sila potiska volkove, ki zaidejo na rob teritorija, proti notranjosti teritorija:

$$\vec{f}_{e,i} = \sum_{e \in E} \frac{\vec{p}_i - \vec{p}_e}{|\vec{p}_e - \vec{p}_i|^2}, \tag{4.7}$$

kjer je E množica vseh robov teritorija, $\vec{p}_i$ trenutna pozicija plenilca in $\vec{p}_e$ najbližja točka roba plenilcu.

Težnja izmikanja oviram

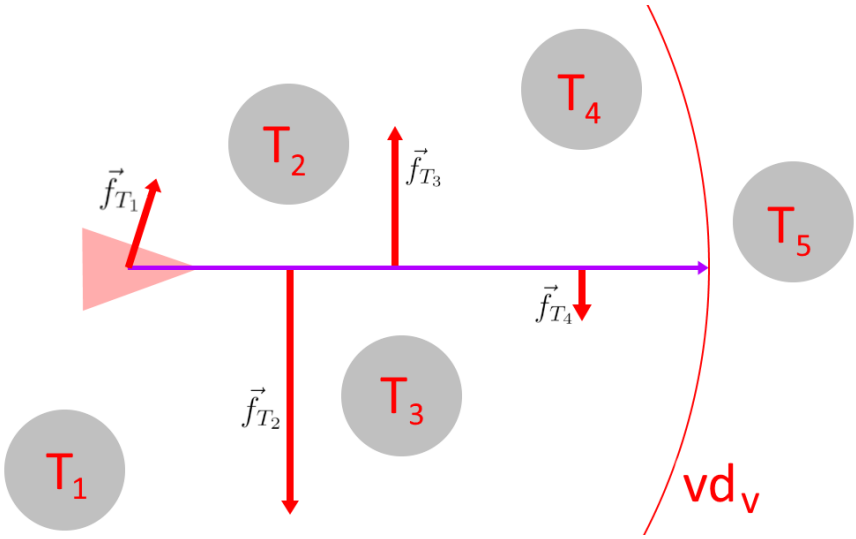
Ker je okolje posejano z raznimi ovirami smo volkovom morali dodati težnjo za izmikanje od njih (slika 4.5).

Tekom simulacije se volk zaveda vseh dreves, ki so znotraj njegovega vidnega polja vd_v. Silo težnje izmikanja oviram $\vec{f}_{oa,i}$ izračunamo tako, da sile posameznih ovir, ki so bližje od vd_v seštejemo in vsoto pomnožimo z -1, zato da smer sile obrnemo stran od ovir:

$$\vec{f}_{oa,i} = -\sum_{i \in T} \vec{f}_{T_i}, \tag{4.8}$$

kjer je T množica vseh ovir.

Slika 4.5: Volk se s pomočjo težnje izmikanja oviram uspešno sprehaja med drevesi. Njegova pot je prikazana s svetlo modro barvo.



Slika 4.6: Vpliv ovir na silo izmikanja oviram.

Silo posameznega drevesa $\vec{f}_{T_i}$ (slika 4.6) dobimo s pomočjo naslednje enačbe:

$$\vec{f}_{T_i} = \overrightarrow{A_i B_i} \cdot (w_{od}^2 - |\overrightarrow{A_i B_i}|^2) \cdot w_i, \tag{4.9}$$

kjer je A_i pozicija ovire, B_i projekcija točke A_i na smerni vektor plenilca, w_{od} konstanta, ki jo nastavimo na začetku simulacije in predstavlja utež oddaljenosti ovire od smernega vektorja plenilca in w_i:

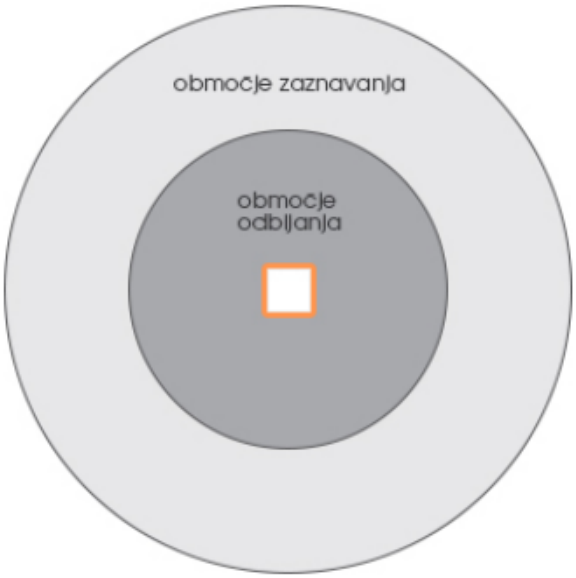
$$w_i = 1 - \frac{od}{vd_v}, \tag{4.10}$$

kjer je *od* razdalja od trenutne pozicije volka do točke B_i in vd_v radij vidnega polja volka. Parameter w_i predstavlja utež oddaljenosti ovire od volka.

4.2 Obkoljevanje plena

Ko volkovi najdejo plen, ga želijo obkoliti. Za implementacijo obkoljevanja smo se zgledovali po modelu narejenem v sklopu diplomske naloge Mitje Goriška [3], ki je zasnovana na podlagi znanstvenega članka avtorjev Escobedo et al. [13].

Gre za model v 3 dimenzionalnem svetu, kjer je vsak posameznik opisan z lokacijo $\vec{p}_i$ in smernim vektorjem gibanja $\vec{v}_i$. Simuliranje vedenja osebkov temelji na območju zaznavanja ostalih osebkov [2]. V Goriškovem modelu obstajata dve območji, ki vplivata na vedenje volkov pri obkoljevanju plena (slika 4.7) [3]. Območje zaznavanja označuje območje v katerem plen vidi plenilca, območje odbijanja pa področje v katerem je plenilec ogrožen, če se frontalno sooči s plenom.



Slika 4.7: Območji, ki vplivata na vedenje volkov pri obkoljevanju plena v modelu [3].

Primarni cilj plenilcev je obkoliti in nato onesposobiti plen. Vsak plenilec se približuje plenu v ravni liniji, neodvisno od ostalih plenilcev. S približevanjem preneha na minimalni varnostni razdalji dc, ki jo definiramo pred začetkom simulacije, in se pomakne stran od ostalih plenilcev, ki so prav tako na razdalji dc.

Premik plenilca v eni sliki simuliramo tako, da njegovi trenutni lokaciji prištejemo vsoto sil, ki delujejo na plenilca (enačba 4.11). To so sila trenja $\vec{f}_{f,i}$, sila privlačnosti plena $\vec{f}_{p,i}$, sila odbojnosti plenilcev $\vec{f}_{j,i}$ in sila odbojnosti vidnega kota plena $\vec{f}_{cone,i}$.

$$\vec{v}_i = \vec{f}_{f,i} + \vec{f}_{p,i} + \vec{f}_{j,i} + \vec{f}_{cone,i}. \tag{4.11}$$

S pomočjo sile trenja $\vec{f}_{f,i}$ lahko spreminjamo vpliv tal na hitrost premika volka. Izračunamo jo po naslednji enačbi:

$$\vec{f}_{f,i} = -C_f \vec{v}_i, \tag{4.12}$$

kjer je C_f koeficient trenja in $\vec{v_i}$ vektor trenutne hitrosti plenilca.

S silo privlačnosti plena $\vec{f}_{p,i}$ modeliramo privlačenje plenilca proti plenu, silo izračunamo kot:

$$\vec{f}_{p,i} = C_W^P \frac{\vec{p}_p - \vec{p}_i}{|\vec{p}_i - \vec{p}_p|^2} \left(1 - \frac{dc^2}{|\vec{p}_i - \vec{p}_p|^2}\right), \tag{4.13}$$

kjer sta C_W^P in dc konstanti, ki ju nastavimo na začetku simulacije. Vektor $\vec{p}_p$ predstavlja trenutno lokacijo plena, $\vec{p}_i$ pa trenutno lokacijo opazovanega plenilca.

Odmik opazovanega plenilca od ostalih plenilcev simuliramo s pomočjo sile odbojnosti plenilcev $\vec{f}_{j,i}$. Za izračun sile $\vec{f}_{j,i}$ seštejemo vpliv vseh ostalih plenilcev na opazovanega plenilca:

$$\vec{f}_{j,i} = -\sum_{j=1, j\neq i}^{N} C_W^W \frac{\vec{p}_j - \vec{p}_i}{|\vec{p}_i - \vec{p}_j|^2} \theta_{j,i}, \tag{4.14}$$

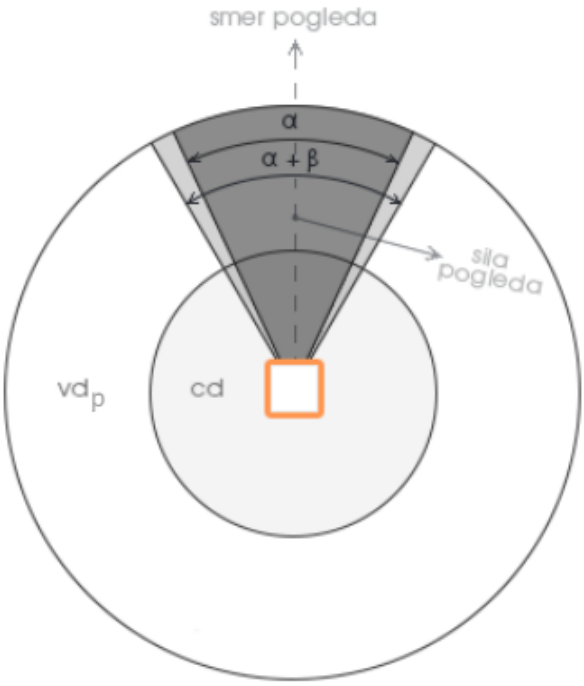
kjer je C_W^W koeficient odbojnosti med plenilci, N množica vseh plenilcev in Gaussova funkcija $\theta_{j,i}$:

$$\theta_{j,i} = exp(-cw((R_{i,p} - da)^2 + (R_{j,p} - da)^2)), \qquad j \in N, j \neq i, \tag{4.15}$$

kjer je da kritična razdalja, $R_{i,p}$ razdalja od opazovanega plenilca do njegovega plena in $R_{j,p}$ razdalja od ostalih plenilcev do plena, ki ga napada opazovani plenilec. Konstanta cw pa predstavlja širino Gaussove funkcije.

Gorišek [3] je parameter cw uporabil kot utež posameznega volka pri simuliranju alfa para (alfa samec in alfa samica). Čeprav ni dokazano, da v divjini alfa pari obstajajo [8], pa lahko parameter cw uporabimo kot prikaz fizične moči in vpliva posameznega volka. Ta parameter namreč povzroči, da se volkovi z višjo vrednostjo cw bolj približajo plenu, kar posledično pomeni da ga tudi večkrat napadejo.

S silo odbojnosti vidnega kota plena $\vec{f}_{cone,i}$ simuliramo izmikanje plenilcev vidnemu polju plena in s tem povečamo njihove možnosti za uspešen napad.



Slika 4.8: Vidni kot plena in prikaz sile pogleda, ki vpliva na plenilca [3].

Vidni kot plena je sestavljen iz kota α, odmika kota β in dolžine vidnega polja vd_p, kot je prikazano na sliki 4.8. Sila $\vec{f}_{cone,i}$ vpliva na plenilca, kadar se ta nahaja znotraj vidnega polja plena.

Silo odbojnosti vidnega kota plena $\vec{f}_{cone,i}$ izračunamo po enačbi:

$$\vec{f}_{cone,i} = C_C^P \cdot \vec{d} \cdot cone_{fac}, \tag{4.16}$$

kjer je C_C^P koeficient vidnega kota, $\vec{d}$ vektor smeri pogleda in $cone_{fac}$ faktor vpliva na silo vidnega kota. Pove nam kako močno plenilca sila potiska vstran od centra vidnega kota glede na njegovo trenutno pozicijo:

$$cone_{fac} = \left(1.5 - \frac{\gamma}{\frac{\alpha}{2} + \beta}\right), \tag{4.17}$$

kjer je γ trenutni kot med plenilcem in centrom vidnega kota. Izračunamo ga po kosinusnem izreku.

V modelu Goriška smo po naših ocenah zasledili več pomankljivosti, ki model oddaljujejo od realnega sveta. Ker je naša želja čim bolj realna simulacija smo te pomankljivosti odpravili s pomočjo nadgradnje določenih pravil obkoljevanja oziroma dodajanjem novih pravil.

Pravilo najbližjega plena

Gorišek se je osredotočil predvsem na primere, kjer trop volkov obkroža in napada en sam izoliran plen. V našem primeru je plenov več. Zato določimo novo pravilo: če plenilec pozna lokacijo več kot enega plena, si za tarčo napada izbere najbližjega. Seveda pa je potrebno določiti tudi neko toleranco menjave tarče za bolj realističen prikaz lova. Volk spremeni svojo tarčo, v primeru ko velja pravilo:

$$R_{current} - R_{closest} > changeTargetPreyTolerance, \tag{4.18}$$

kjer je $R_{current}$ razdalja od plenilca do trenutne tarče, $R_{closest}$ razdalja od plenilca do najbližjega plena in $changeTargetPreyTolerance$ konstanta, ki nam pove kdaj zamenjati tarčo.

Nevidnost volka

Volk je plenu neviden v primeru ko je med volkom in plenom neprehodna ovira (npr. drevo), se nahaja v visoki travi ali ko ni v vidnem polju plena.

Ker v realnosti plen ne izgubi sledi za plenilcem v istem trenutku, ko volk izgine iz njegovega vidnega polja, dodamo toleranco vidnosti volka $invisibilityTolerance$. S tem odpravimo možnost, da bi plen izgubil sled za plenilcem takoj ko ta vstopi v visoko travo ali zaide za drevo, kar bi povzročilo preveč robotsko obnašanje plena in plenilcev. Volk postane neviden, če velja pravilo:

$$t_s - t_{invisible} > invisibilityTolerance, \tag{4.19}$$

kjer je t_s čas simulacije in $t_{invisible}$ čas, ko volk vstopi v polje nevidnosti.

Podobno dodamo še toleranco vidnosti volka po zadanem udarcu plenu $visibilityAfterHitTolerance$. S tem skušamo dodati nekaj nevarnosti volkovom tudi v primeru, ko napadajo plen, ki je zašel v visoko travo. Volk je po zadanem udarcu plenu viden dokler velja pravilo:

$$wolfRetreat \ \ \& \ \ (t_s - t_{visible} < visibilityAfterHitTolerance), \tag{4.20}$$

kjer je $wolfRetreat$ spremenljivka, ki nam pove ali je volk v stanju umika od plena po zadanem udarcu in $t_{visible}$ čas ko je volk postal viden plenu oz. ko mu je zadal udarec.

Težnja izmikanja oviram

Tako kot pri zasledovanju plena, se morajo volkovi izmikati oviram iz okolja tudi pri obkoljevanju. Zato k izračunu končnega premika $\vec{v}_i$ dodamo še silo težnje izmikanja oviram $\vec{f}_{oa,i}$:

$$\vec{v}_i = \vec{f}_{f,i} + \vec{f}_{p,i} + \vec{f}_{j,i} + \vec{f}_{cone,i} + \vec{f}_{oa,i}, \tag{4.21}$$

kjer silo $\vec{f}_{f,i}$ izračunamo po enačbi (4.12), silo $\vec{f}_{p,i}$ po enačbi (4.13), silo $\vec{f}_{j,i}$ po enačbi (4.14) in silo $\vec{f}_{cone,i}$ po enačbi (4.16). Silo težnje izmikanja oviram $\vec{f}_{oa,i}$ pa izračunamo po enačbi (4.9). Za razliko od plena, se volkovi visoki travi ne izmikajo, saj jim daje prednost pri lovu.

Vzdržjivost

Da bi prikazali volkove kot živa bitja z realnimi zmogljivostmi, jim dodamo še vzdržljivost. Vzdržljivost vpliva tako na hitrost premikanja volkov po teritoriju, kot tudi na hitrost napada.

Vzdržljivost je predstavljena kot število med 0 in 1, s katerim pomnožimo premike volkov. Na začetku simulacije je nastavljena na 1, znižuje pa se medtem ko se volk premika po teritoriju, napada ali ko prejme udarec.

Stopnja znižanja vzdržljivosti v vseh treh primerih je nastavljiva s parametri. V principu pa se volku vzdržljivost zniža največ, ko prejme udarec in najmanj ko se prosto premika po teritoriju.

Odprava pomankljivosti sile odbojnosti vidnega kota plena

Pri testiranju Goriškovega modela [3] smo ugotovili pomankljivost sile odbojnosti vidnega kota plena $\vec{f}_{cone,i}$ v povezavi z našo implementacijo obrambe plena. Če plen dlje časa gleda naravnost v plenilca, se plenilec ustavi, saj se ne more odločiti v katero smer naj se izmika pogledu plena.

V Goriškovem modelu je sila $\vec{f}_{cone,i}$ delovala zadovoljivo zaradi preprostosti implementacije obrambe plena. Rotacija plena je v njegovem modelu implementirana tako, da se volk med izvajanjem rotacije ne zaveda sprememb v okolju. Plen v nekem trenutku zazna kateri plenilec mu je najbližje in si zapomni njegovo trenutno lokacijo. Začne se obračati proti lokaciji, ki si jo je zapomnil. Če se ta plenilec tekom rotacije plena premakne ali pa najbližji postane drug plenilec, se plen tega ne bo zavedal. Ko plen dokonča rotacijo ponovi postopek iskanja najbližjega plenilca in rotacije proti njemu. Ker pa na plenilca deluje sila odbojnosti vidnega kota $\vec{f}_{cone,i}$, le-ta tekom rotacije plena nikoli ne bo ostal na istem mestu. Posledica tega je, da nikoli ne pride do situacije, ko bi bil plen usmerjen naravnost proti plenilcu za več časa.

To težavo smo odpravili tako, da smo vsakemu volku nastavili levo oziroma desno naravnanost. V primeru, ko bi se zgodilo, da plen gleda naravnost v volka, se bo volk glede na svojo naravnanost začel izmikati v levo oziroma desno stran okoli plena z rahlim oddaljevanjem od plena. To pripelje do veliko boljših rezultatov in uspešnega izmikanja plenilcev pogledu plena, tudi v prej opisanem primeru.

Z uporabo te rešitve teoretično lahko pride do nerealističnega dolgotrajnega kroženja plenilca okoli plena. To se lahko zgodi, če je v obkoljevanje vključen samo en plenilec in v okolici, kjer se izvaja obkoljevanje, ni naravnih ovir, kamor bi se ta plenilec lahko skril. Zaradi narave simulacije, ki predpostavlja več kot enega plenilca in dinamično okolje z ovirami, pa je to

dovolj dobra rešitev problema. V večini primerov bo plen namreč spremenil tarčo rotacije ali izgubil sled za plenilcem preden bi prišlo do nerealističnega obnašanja volka.

4.3 Napad

Do napada na plen pride, ko je volk v stanju obkoljevanja plena in oceni, da lahko plen varno napade. Plen, ki se bori za svoje življenje lahko svoje napadalce tudi poškoduje. Tega se zavedajo tudi volkovi, zato je njihov cilj plenu zadati udarec, ko se le-ta ne more braniti. Če napad plena ne ubije se plenilec umakne nazaj na varno razdaljo, kjer zopet čaka na primeren trenutek za ponovni udarec.

V Goriškovem modelu [3] je napad implementiran tako, da se plenilec skuša približati plenu, dokler ni znotraj območja odbijanja. Če plenilec pride v območje odbijanja in ni v vidnem polju plena ter je od zadnjega udarca minilo več kot predviden čas $hitFrequency$, plenilec zada udarec plenu. Če pa pride v območje odbijanja in je v vidnem polju plena, obstaja možnost, da ga bo plen udaril. V primeru prejetega udarca, se umakne na varno razdaljo za določen čas $dc_{penalty}$, ki se ga izračuna po enačbi:

$$dc_{penalty} = dc + 0.3 \cdot n_{hit} \cdot m_p \cdot \left(\frac{time\_penalty\_left}{time\_penalty} \right), \tag{4.22}$$

kjer je dc prvotna varnostna razdalja, n_{hit} število prejetih udarcev plenilca in m_p masa plena.

Način napada smo za potrebe našega modela spremenili, da bi ustvarili bolj dinamično okolje. Pomankljivost napada iz [3] je v tem, da medtem, ko je en izmed plenilcev napadal, drugi niso uspeli priti blizu plena zaradi odbojne sile ostalih plenilcev $\vec{f}_{p,i}$. Poleg tega bi pri uporabi načina napada iz [3] v našem primeru prišlo do velikega števila prejetih udarcev na strani volkov. To pa bi pomenilo preveliko odstopanje končnih rezultatov naše od končnih rezultatov Goriškove simulacije [3].

V našem primeru zato volk napade plen, če:

– ni ranjen,

– ni v stanju umika na varno razdaljo dc po zadanem udarcu,

– je izven vidnega polja plena,

– je, odkar je plen zadnjič videl volka, minilo več kot predviden čas t_{ir}.

Oziroma, če velja pravilo:

$$!isHit \ \& \ !wolfRetreat \ \& \ isInvisible \ \& \ (t_s - t_{ls} > t_{ir}), \tag{4.23}$$

kjer je $isHit$ spremenljivka, ki nam pove ali je volk ranjen, $wolfRetreat$ spremenljivka, ki nam pove ali je v stanju umikanja na varno razdaljo, $isInvisible$ spremenljivka, ki nam pove ali je volk plenu neviden, t_s čas simulacije in t_{ls} čas ko je plen zadnjič videl volka.

Volk zada udarec plenu, ko pride na določeno razdaljo od plena $wolfReach$. Udarec plenu zniža življensko energijo in vzdržljivost. Škoda, ki jo plenilec zada z enim udarcem, je odvisna od vrednosti parametra cw, ki določa fizično moč in vpliv plenilca. Po zadanem udarcu se volk umakne na varno razdaljo dc in ponovi proces napada, če so pogoji za napad zadoščeni.

V primeru, da plenilec prejme udarec, se mora umakniti na novo varno razdaljo $dc_{penalty}$. Prejeti udarec plenilcu zniža vzdržljivost za prednastavljeno vrednost.

Poglavje 5

Vedenje plena

Kot primer plena v našem modelu smo vzeli večjo žival, ki se je sposobna agresivno braniti pred plenilci ter je sposobna odbiti večje število napadov volkov. V naravi tak scenarij najdemo ko skupina volkov napada bizona ali bivola [14]. Izhodiščni model [3] je pri simulacijah uporabljal samo en plen, za potrebe naših simulacij smo model nadgradili, tako da zdaj podpira poljubno število plenov.

5.1 Čredenje in sprehajanje

Ker smo model nadgradili tako, da v simulaciji lahko prikažemo poljubno število plenov, smo plenom dodali še credni nagon. Z nastankom čred ima posamezen plen boljše možnosti za preživetje volčjih napadov, oziroma lahko preživi dlje kot če bi bil sam. V primeru, če volkovi obkolijo čredo, se bo vsak plen branil pred sebi najbližjim plenilcem. Posledično si bodo pleni v čredi drug drugemu krili hrbet in s tem povečali možnost uspešne obrambe pred napadalci.

Čredenje in sprehajanje smo implementirali na podoben način kot pri volkovih. Zgledovali smo se po Reynoldsu [2] in Demšarju [9]. Če plen ne vidi nobenih volkov, se le-ta naključno giba po območju. Medtem želi zadovoljiti težnje kohezije, poravnave in razmika. Zaradi specifičnosti problema pa smo

pristop seveda morali do določene mere prilagoditi in nadgraditi. Ker je logika čredenja in sprehajanja plenov zelo podobna logiki sprehajanja volkov iz poglavja 4.1, bomo opisali samo razlike med njima.

Cilj plena ni samo sprehajanje mimo ovir, ampak tudi iskanje prostega območja brez ovir. Zato nanj vplivajo vse ovire (drevesa in trava) znotraj njegovega vidnega polja. Plenu silo izmikanja oviram $\vec{f}_{oa,p}$ izračunamo kot:

$$\vec{f}_{oa,p} = \sum_{t \in T} \frac{\vec{p}_p - \vec{p}_t}{|\vec{p}_t - \vec{p}_p|^2} + w_{tg} \cdot \sum_{g \in G} \frac{\vec{p}_p - \vec{p}_g}{|\vec{p}_g - \vec{p}_p|^2}, \tag{5.1}$$

kjer je T množica vseh dreves, G množica vseh objektov visoke trave, $\vec{p}_p$ trenutna lokacija opazovanega plena, $\vec{p}_t$ lokacija drevesa in $\vec{p}_g$ lokacija visoke trave. Vsoto sil visoke trave množimo s konstanto w_{tg}, ki jo določimo glede na velikost trave, ki je uporabljena v modelu. V našem primeru smo uporabili $w_{tg} = 0{,}1$.

Plenom ni potrebno preiskovati teritorija, zato pri izračunu premika plena ne potrebujemo sile gibanja proti cilju. Posledično prevlada vpliv sile izmikanja oviram $\vec{f}_{oa,p}$ in tako dosežemo navidezno iskanje prostega območja (brez nevarnih objektov, ki bi lahko zmanjšali učinkovitost obrambe plena).

Če bi simulacijo razširili in definirali večje okolje simulacije, bi lahko izpustili tudi silo robov $\vec{f}_{oa,p}$. S tem bi plenom dovolili zapustiti teritorij volkov, volkovi pa bi ostali brez vira hrane. Zgolj zato, ker želimo simulacijo izpeljati do konca (dokler en izmed plenov ni onesposobljen), pa to silo ohranimo.

5.2 Branjenje

Ker smo definirali plen kot žival, ki se je sposobna braniti pred napadalci (bizon, bivol...), mu moramo nadgraditi tudi logiko obrambe. Način obrambe iz Goriškovega modela [3] namreč, zaradi svoje enostavnosti, ne zadostuje za prikaz realistične samoobrambe plena.

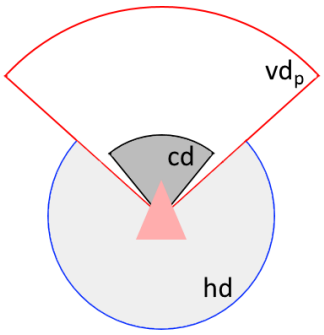
Obramba je v modelu [3] implementirana tako, da plen v vsakem trenutku pozna lokacije vseh napadalcev znotraj območja zaznavanja vd_p, zato

se vedno poskuša obrniti proti tarči, ki ji je najbližja. Če pa plen zazna plenilca znotraj območja branjenja cd in je od zadnje obrambe minilo več kot predviden čas $hitFrequency$, plen udari plenilca.

Pred začetkom simulacije lahko plenu nastavimo poljubno količino življenjske energije. Ob izgubi celotne življenjske energije je plen onesposobljen.

Tako kot obnašanje volkov, želimo tudi obnašanje plena približati realnosti, zato plenu spremenimo način samoobrambe. Plen zaznava napadalce s pomočjo sluha in vida. Brani pa se pred njemu najbližjim plenilcem tako, da se obrne proti plenilcu in se vanj agresivno zaleti. V primeru, da ga določen plenilec udari, si ga plen nastavi za naslednjo tarčo.

Pri plenu ločimo tri območja zaznavanja (slika 5.1): območje vidnega zaznavanja vd_p, območje slušnega zaznavanja hd in območje kritičnega zaznavanja cd.



Slika 5.1: Območja zaznavanja plena.

Območje vidnega zaznavanja vd_p sega najdlje in ima največji vpliv na obrambo plena. Če plen zazna napadalce znotraj vd_p, ignorira napadalce, ki bi jih morda zaznal v območju slušnega zaznavanja hd in se brani samo pred plenilci v svojem vidnem polju. Plen se odziva na napadalce v območju slušnega zaznavanja zgolj, če v območju vidnega zaznavanja ni zaznal nobe-

nega plenilca. Če plen zazna napadalca v območju kritičnega zaznavanja *cd* in je od njegovega zadnjega napada minilo dovolj dolgo, napade plenilca.

Območje kritičnega zaznavanja *cd* je podmnožica območja vidnega zaznavanja vd_p. Velikost območij je nastavljiva s parametri, v okviru naše simulacije pa nastavimo parametre tako, da velja $vd_p > hd > cd$.

V Goriškovem modelu je premik plena med samoobrambo modeliran s pomočjo sile pregona v obliki odbojnosti. Premik plena $\vec{v}_p$ je izračunan kot vsota sile pregona $\vec{f}_{p,p}$ in sile trenja $\vec{f}_{f,p}$:

$$\vec{v}_p = \vec{f}_{p,p} + \vec{f}_{f,p}, \tag{5.2}$$

kjer je sila pregona izračunana po enačbi:

$$\vec{f}_{p,p} = -1 \cdot \sum_{i=1}^{N} C_P^W \frac{\vec{p}_p - \vec{p}_i}{|\vec{p}_i - \vec{p}_p|^2}, \tag{5.3}$$

kjer je C_P^W koeficient sile pregona. Seštevek sil plenilcev pomnožimo z -1 zato, da dosežemo premik stran od plenilcev. Silo trenja $\vec{f}_{f,p}$ pa izračunamo po enačbi:

$$\vec{f}_{f,p} = -C_f \vec{v}_p, \tag{5.4}$$

kjer je C_f koeficient sile trenja in $\vec{v}_p$ vektor trenutne hitrosti opazovanega plena.

V našem modelu se plen med branjenjem premika podobno kot v Goriškovem modelu. Ker pa se v našem primeru plen nahaja v simulaciji realnega okolja, se mora med obrambo še vedno izmikati nevarnim objektom (drevesa in visoka trava). Zato enačbi (5.2) dodamo še silo izmikanja oviram $\vec{f}_{oa,p}$:

$$\vec{v}_p = \vec{f}_{p,p} + \vec{f}_{f,p} + \vec{f}_{oa,p}, \tag{5.5}$$

kjer $\vec{f}_{oa,p}$ izračunamo s pomočjo enačbe (5.1).

Pri izračunu sile pregona, v enačbi (5.3) spremenimo množico N tako, da vsebuje samo napadalce, ki trenutno ogrožajo plen. V množico N dodamo

napadalce iz območja vd_p, če pa v območju vd_p ni nobenega plenilca vanjo dodamo napadalce iz območja hd.

Tako približamo obrambo plena stanju v naravi, saj se plen ne bo zavedal nevarnosti, ki mu preži od zadaj ali iz boka, dokler bo osredotočen na obrambo pred plenilcem, ki ga ogroža od spredaj. Prav tako bodo volkovi z obkoljevanjem lažje in uspešnejše ujeli plen kot z direktnim napadom.

Plen se aktivno brani s protinapadi. Napad izvrši, če zazna volka v območju kritičnega zaznavanja cd in že dovolj dolgo ni napadel. Vnaprej ima določeno hitrost napada $attackSpeed$, doseg $preyReach$ in omejen čas napada z $maxAttackTime$. Plen uspešno udari volka, če se mu med napadom približa na $preyReach$ razdaljo. V tem primeru plen napadenega volka poškoduje. Če pa plenu ne uspe udariti volka v času $maxAttackTime$, se vrne nazaj v stanje iskanja najbližjega napadalca in obrambe z rotacijo.

Na enak način kot pri volkovih, je vzdržljivost definirana tudi pri plenu. Ta izgublja vzdržljivost med premikanjem, branjenjem in ob prejemu udarca. Razlika med plenom in volkovi je zgolj v tem, da je pri plenu pomembna hitrost rotacije zaradi samoobrambe. Zato pri plenu vzdržljivost vpliva tudi na hitrost njegove rotacije. S tem povzročimo, da se plenu z manjšanjem vzdržljivosti manjša tudi učinkovitost samoobrambe.

5.3 Dodatna razširitev modela

Po našem mnenju smo z modelom pokrili večji del značilnosti plenjenja volkov v naravi, ter s tem dosegli naš cilj vizualno prepričljivega dogajanja v okviru računalniške igre. Vendar pa so primeri, ko bi se krdelo volkov spopadlo z močnim odraslim plenom, v naravi zelo redki. Volkovi znajo zelo dobro oceniti uspešnost napada in se zavedajo posledic napada na tak plen [1]. Ker je tveganje poškodb pri takšnem napadu preveliko, so v večini primerov tarče volkov oslabljenje živali in mladiči, saj se le-ti niso sposobni braniti in volkovom predstavljajo manjšo nevarnost [4].

Naš model plenjenja nadgradimo za prikaz, v naravi pogostejšega, načina

Slika 5.2: Primer napada na čredo z mladiči.

plenjenja volkov. Volkovi si za tarčo napada primarno izbirajo mladiče. Odrasle živali bodo napadli le v primeru, ko mladičev ne zaznajo. Mladiči so šibkejši od odraslih, zato jih volkovi lahko hitreje onesposobijo. Mladiči prav tako nimajo možnosti obrambe pred volkovi, zato se za obrambo zanašajo na odrasle. V primeru ogroženosti se mladiči zatečejo k najbližjemu odraslemu, ki je znotraj njihovega vidnega polja (slika 5.2). Pri mladičih zato nadgradimo enačbo (5.5) tako, da prištejemo silo težnje po bližini odraslega $\vec{f}_{a,y}$:

$$\vec{v}_p = \vec{f}_{p,p} + \vec{f}_{f,p} + \vec{f}_{oa,p} + \vec{f}_{a,y}, \tag{5.6}$$

kjer $\vec{f}_{a,y}$ izračunamo po enačbi:

$$\vec{f}_{a,y} = \frac{\vec{p}_a - \vec{p}_y}{|\vec{p}_a - \vec{p}_y|} \cdot \frac{R_{a,y}}{vd_y}, \tag{5.7}$$

kjer je $\vec{p}_a$ pozicija mladiču najbližjega odraslega, $\vec{p}_y$ pozicija opazovanega mladiča, $R_{a,y}$ razdalja od opazovanega mladiča do njemu najbližjega odraslega in vd_y dolžina vidnega polja opazovanega mladiča.

Na podoben način dodamo tudi odraslim plenom težnjo po varovanju mladičev. Če so v vidnem polju posameznega odraslega plena tudi mladiči, se le-ta v primeru zaznane nevarnosti, pomakne proti sebi najbližjemu mladiču in ga skuša zavarovati pred napadalci.

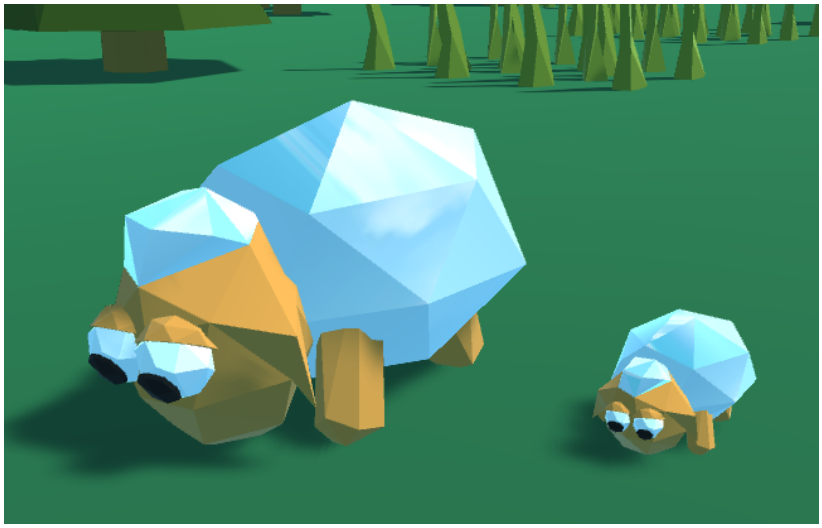
Poglavje 6

Vizualizacija in analiza

6.1 Vizualizacija

Da bi simulacijo naredili čim bolj intuitivno in jo približali ljudem, jo vizualiziramo v 3D okolju. Za vizualizacijo okolja smo uporabili modele drevesa in trave iz projekta FRI Sheeping [15] (slika 5.2). Iz istega projekta smo uporabili tudi animirane modele za prikaz plena in plenilcev. Za plenilce smo uporabili model psa (slika 6.1), za prikaz plena pa smo uporabili model ovce (slika 6.2).

Slika 6.1: Model plenilca iz projekta [15].



Slika 6.2: Model odraslega plena in mladiča iz projekta [15].

6.2 Analiza rezultatov simulacije

Da bi ugotovili, kako uspešen je model pri imitiranju dogajanja v naravi, smo rezultate simulacije primerjali s prosto dostopnimi video posnetki s spleta [14, 16].

Slika 6.3: Protinapad odraslega plena, da bi zaščitil mladiča.

Na sliki 6.3 vidimo podobno obrambo plena v naravi in v naši simulaciji. Plenilec napade mladiča, ko odrasel plen ni pozoren. Ko odrasel plen zazna, da je mladič v nevarnosti, se takoj obrne in agresivno napade plenilca.

Slika 6.4: Plenilci napadejo plen, ko ta ni pozoren.

Na sliki 6.4 vidimo, da se plenilec skuša približati mladiču medtem ko je odrasel plen osredotočen na drugega plenilca. Po zaznani grožnji se odrasel plen obrne proti plenilcu. Plenilec se umakne na varno razdaljo, medtem pa se izmika pogledu odraslega plena. Po primerjavi ugotovimo, da smo se dogajanju v naravi približali dovolj dobro za uporabo modela v računalniških igrah. V simulaciji uspemo prikazati ključne lastnosti plenjenja volkov, medtem ko se izognemo pretiranemu zapletanju modela. Zaradi tega model ni primeren za raziskovalne in znanstvene namene, ampak služi kot enostaven prikaz fenomena, ki pa je dovolj natančen za uporabo v zabavni industriji.

Med testiranjem smo beležili podatke o poteku simulacije. Zanimalo nas je kako na uspešnost plenjenja vpliva velikost krdela in prisotnost naravnih ovir.

Velikost krdela	Čas (s)	Standardni odklon
3	47,60	26,46
5	26,64	5,39
7	15,60	3,50
11	17,98	4,13

Tabela 6.1: Povprečen čas plenjenja v okolju brez naravnih ovir, ob napadu na en odrasel plen in njegovega mladiča.

Velikost krdela	Poškodbe	Standardni odklon
3	1,92	1,41
5	0,94	0,76
7	0,13	0,33
11	0,32	0,49

Tabela 6.2: Povprečno število poškodb na volka v okolju brez naravnih ovir, ob napadu na en odrasel plen in njegovega mladiča.

V tabelah 6.1, 6.2, 6.3 in 6.4 so prikazana povprečja rezultatov simulacije ob napadu krdela na en odrasel plen in njegovega mladiča. Za vsako kombinacijo smo izvedli 25 ponovitev. Med seboj smo primerjali krdela s 3, 5, 7 in 11 volkovi. Čas začnemo meriti, ko volkovi začnejo z obkoljevanjem.

Iz podatkov v tabelah 6.1 in 6.2 lahko razberemo, da je v primeru, ko se simulacija odvija na prostem (brez dreves in trave), najbolj učinkovito krdelo s 7 ali več volkovi. To krdelo namreč v najkrajšem času onesposobi plen, poleg tega pa je tudi pričakovano število poškodb volkov najmanjše.

Velikost krdela	Čas (s)	Standardni odklon
3	47,03	19,08
5	27,60	6,08
7	20,32	6,09
11	20,08	4,05

Tabela 6.3: Povprečen čas plenjenja v okolju z naravnimi ovirami, ob napadu na en odrasel plen in njegovega mladiča.

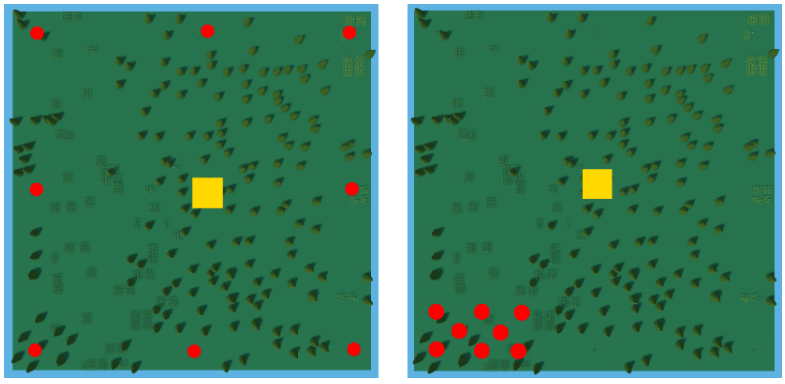
V tabelah 6.3 in 6.4 so prikazani podatki, ki smo jih zabeležili po tem ko smo v simulacijo dodali naravne ovire (drevesa in travo). V primerjavi s tabelama 6.1 in 6.2 se čas plenjenja in pričakovano število poškodb na volka, pri krdelih z več volkovi, rahlo povečata. Do tega pride zaradi manjšega manevrskega prostora volkov in nepopolne obkolitve plena. Plen ima tako večje možnosti odbijanja volkov. Prednosti okolja z drevesi in travo pridejo do

Velikost krdela	Poškodbe	Standardni odklon
3	1,69	1,21
5	0,91	0,78
7	0,56	0,70
11	0,44	0,57

Tabela 6.4: Povprečno število poškodb na volka v okolju z naravnimi ovirami, ob napadu na en odrasel plen in njegovega mladiča.

izraza pri krdelih z manj volkovi, kjer se čas plenjenja in verjetnost poškodb rahlo zmanjšata. Čeprav je krdelo z 11 volkovi prej ujelo plen in je verjetnost poškodbe posameznega volka manjša kot pri krdelu s 7 volkovi, je krdelo s 7 volkovi še vedno bolj učinkovito. Upoštevati moramo namreč tudi dejstvo, da bo posamezen volk v večjem krdelu dobil manjši delež hrane od ulova, kot volk v manjšem krdelu.

Z analizo podatkov, ki smo jih zabeležili med testiranjem simulacij smo ugotovili zakaj v naravi največkrat najdemo krdela, ki štejejo okoli 7 volkov. Razmerje med koristjo in ceno lova je v tem primeru najboljše. Poleg tega smo ugotovili, da prisotnost naravnih ovir negativno vpliva na izid lova pri večjih krdelih in pozitivno pri manjših krdelih.



Slika 6.5: Začetni postavitvi volkov za testiranje celotne simulacije, kjer so volkovi obarvani z rdečo plen pa z rumeno barvo.

Začetna postavitev	Povprečen čas (s)	Standardni odklon
skupinska	156,51	141,62
posamezno	120,44	74,40

Tabela 6.5: Povprečen čas iskanja in plenjenja ter standardni odklon v okviru celotne simulacije.

Začetna postavitev	Poškodbe	Standardni odklon
skupinska	0,57	0,71
posamezno	0,64	1,14

Tabela 6.6: Povprečno število poškodb volkov in standardni odklon v okviru celotne simulacije.

V tabeli 6.5 je prikazan povprečen čas, v tabeli 6.6 pa povprečno število poškodb volkov v okviru celotne simulacije (iskanje plena in plenjenje). Rezultate smo beležili za dve različni postavitvi volkov (slika 6.5). V prvem primeru volkovi začnejo iskanje v že formiranem krdelu, v drugem pa začne vsak volk preiskovati teritorij sam. Za vsako postavitev smo izvedli 80 ponovitev.

Iz tabele 6.5 je razvidno, da je čas, ki ga volkovi potrebujejo za lov, zelo nepredvidljiv. Poleg tega na podlagi podatkov iz tabel 6.5 in 6.6 ugotovimo, da ima vsaka postavitev svoje prednosti in slabosti. V primeru skupinske začetne postavitve, celoten proces iskanja plena in nato plenjenja, traja dlje kot če so na začetku volkovi raztreseni po celotnem teritoriju. Volkovi namreč hitreje najdejo plen, če niso vezani v skupine. Na drugi strani pa volkovi, ki se združijo v skupine hitreje in lažje onesposobijo plen, ko ga najdejo. Posledično utrpijo manj poškodb.

Poglavje 7

Sklepne ugotovitve

V prvem delu diplomske naloge smo na podlagi Reynoldsa [2] in Demšarja [9] implementirali algoritem Boids. Uporabili smo ga za preiskovanje teritorija pri plenilcih in sprehajanje pri plenih. Nato smo ga nadgradili z dodatnimi težnjami in ga tako približali razmeram v naravi. Plenu smo dodali tudi vonj, katerega plenilci lahko zaznajo in tako lažje najdejo plen. S pomočjo algoritma Boids smo dosegli tako združevanje volkov v krdela, kot tudi združevanje plenov v črede.

V drugem delu smo implementirali logiko obkoljevanja in napada plenilcev ter obrambe plena. Za osnovo smo vzeli Goriškov model [3] in ga nadgradili. Ker smo želeli prikazati čim bolj dinamično plenjenje smo spremenili način napada plenilcev tako, da se morajo po zadanem udarcu umakniti od plena. Tako dajo možnost napada tudi ostalim plenilcem. Poleg tega smo spremenili tudi način obrambe plena. Obrambo plena smo nadgradili tako, da se plen agresivno brani pred plenilci, ki ga ogrožajo, z zaletavanjem vanje. Ker smo želeli model približati realnosti in volkovi v realnosti le redko napadejo plen, ki se je sposoben agresivno braniti, smo v simulacijo dodali še mladiče plenov. Volkovi tako prioritizirajo plenjenje mladičev, medtem ko se še vedno izmikajo proti-napadom odraslih plenov.

Simulacijo smo primerjali s prosto dostopnim video materijalom na spletu (npr. video vsebini [14] in [16]) in ugotovili, da smo se realnosti približali

dovolj dobro ter bi model lahko uporabili v računalniških igrah. Če bi želeli simulacijo uporabiti v znanstvene namene, bi bilo potrebno posameznikom znotraj modela dodati še neko vrsto učenja. Tako bi močno izboljšali umetno inteligenco posameznikov in morda dosegli popolen realizem. Zanimivo bi bilo videti, ali bi se volkovi naučili izkoristiti svojo nevidnost v travi oziroma za drevesi.

Med izdelavo diplomskega dela smo razširili svoje znanje o igralnem pogonu Unity. Pridobili smo nova znanja s področja računalniških simulacij in ustvarjanja umetnega življenja (angl. *artificial life*). Prav tako pa smo dobro preučili literaturo s področja biologije, o življenju in plenjenju volkov v divjini, kar nam je simulacijo pomagalo približati stanju v naravi.

Obstaja še veliko možnosti nadgradnje modela. V nadaljevanju bi bilo zelo zanimivo model prenesti v 3D, kjer bi lahko okolje naredili bolj razgibano, z dolinami in hribi. Okolje bi prav tako lahko razširili in dodali vanj nove naravne ovire, kot na primer reke ali sneg. Prav tako bi lahko vizualno nadgradili model. V našem modelu smo se osredotočili predvsem na plenjenje večjih živali, ki so se sposobne braniti ali pa imajo ob sebi druge živali, ki jih branijo. Zelo zanimivo bi bilo simulirati lov na hiter plen, ki se ne brani ampak samo beži pred volkovi in se za preživetje zanaša na svojo agilnost (npr. zajec).

Literatura

[1] L David Mech and Luigi Boitani. *Wolves: behavior, ecology, and conservation*. University of Chicago Press, 2010.

[2] Craig W Reynolds. Flocks, herds and schools: A distributed behavioral model. In *ACM SIGGRAPH computer graphics*, volume 21, pages 25–34. ACM, 1987.

[3] Mitja Gorišek. Krdelo volkov: modeliranje in simulacija skupinskega plenjenja. Diplomska naloga, Univerza v Ljubljani, 2017.

[4] Daniel R Stahler, Douglas W Smith, and Debra S Guernsey. Foraging and feeding ecology of the gray wolf (canis lupus): lessons from yellowstone national park, wyoming, usa. *The Journal of nutrition*, 136(7):1923S–1926S, 2006.

[5] Unity. Dosegljivo: `https://unity3d.com/unity`. [Dostopano: 20. 8. 2018].

[6] Włodzimierz Jedrzejewski, Krzysztof Schmidt, Jörn Theuerkauf, Bogumiła Jedrzejewska, and Rafał Kowalczyk. Territory size of wolves canis lupus: linking local (białowieża primeval forest, poland) and holarctic-scale patterns. *Ecography*, 30(1):66–76, 2007.

[7] Andrew M Kittle, Morgan Anderson, Tal Avgar, James A Baker, Glen S Brown, Jevon Hagens, Ed Iwachewski, Scott Moffatt, Anna Mosser, Brent R Patterson, et al. Wolves adapt territory size, not pack size to local habitat quality. *Journal of Animal Ecology*, 84(5):1177–1186, 2015.

[8] L David Mech. Alpha status, dominance, and division of labor in wolf packs. *Canadian Journal of Zoology*, 77(8):1196–1203, 1999.

[9] Jure Demšar. Primerjava algoritmov za iskanje metričnih sosedov pri simulacijah skupin živih bitij. *Elektrotehniski Vestnik*, 84(5):214–220, 2017.

[10] George B Rabb, Jerome H Woolpy, and Benson E Ginsburg. Social relationships in a group of captive wolves. *American zoologist*, 7(2):305–311, 1967.

[11] Craig W Reynolds. Steering behaviors for autonomous characters. In *Game developers conference*, volume 1999, pages 763–782. Citeseer, 1999.

[12] Fernando Bevilacqua. Understanding Steering Behaviors: Wander. Dosegljivo: `https://gamedevelopment.tutsplus.com/tutorials/understanding-steering-behaviors-wander--gamedev-1624`, 2012. [Dostopano: 28. 7. 2018].

[13] R Escobedo, C Muro, L Spector, and RP Coppinger. Group size, individual role differentiation and effectiveness of cooperation in a homogeneous group of hunters. *Journal of the Royal Society Interface*, 11(95):20140204, 2014.

[14] Discovery. Bison and Her Calf Battle Wolves | North America. `https://www.youtube.com/watch?v=GtG-9ftqoHw`, Junij 2013.

[15] FRI Sheeping. Dosegljivo: `https://connect.unity.com/p/fri-sheeping`. [Dostopano: 20. 8. 2018].

[16] BBC Earth. Wolves vs Herd of Muskox | Snow Wolf Family And Me | BBC Earth. `https://www.youtube.com/watch?v=PkWVGf2sfiw`, Junij 2016.