VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Simulace davu

Semestrální projekt

Vypracoval:
Pavel Drábek

Vedoucí práce:
Ing. Martin Němec, Ph.D.

2017

Obsah

1 Úvod

Simulace davu má za cíl věrně napodobit chování velké množiny objektů. Zdánlivě složité chování však může být dosaženo definováním několika jednoduchých pravidel každého jedince. Chování samotného jedince se může jevit jako zmatené, avšak velká skupina stejně naprogramovaných jedinců může vykazovat inteligentní chování.

Takové chování uplatňujeme nejen ve filmovém a herním průmyslu, ale i v architektuře, nácviku vojenských strategií, navrhování evakuačních plánů, simulování požárních poplachů či chování robotů. [1] Díky moderním technologiím můžeme napodobit chování davu, které by nás stálo nejen velké množství času, ale i finančních prostředků. Například filmová bitva velkých armád, stadion plný fanoušků apod. Můžeme však vytvořit i takové chování, se kterým bychom se v reálném životě nikdy nesetkali - ve filmu Drákula: Neznámá legenda (2014) ovládá hlavní postava desetitisíce netopýrů. [2]

Takovéto systémy většinou vykazují chaotické chování, protože nepatrná změna hodnoty jednoho parametru byť jen zaokrouhlením bude mít dopad na pozdější výsledek. Nicméně dá se zajistit i chování, které pro stejnou situaci chaotické chování nevykazují.

1.1 Cíl projektu

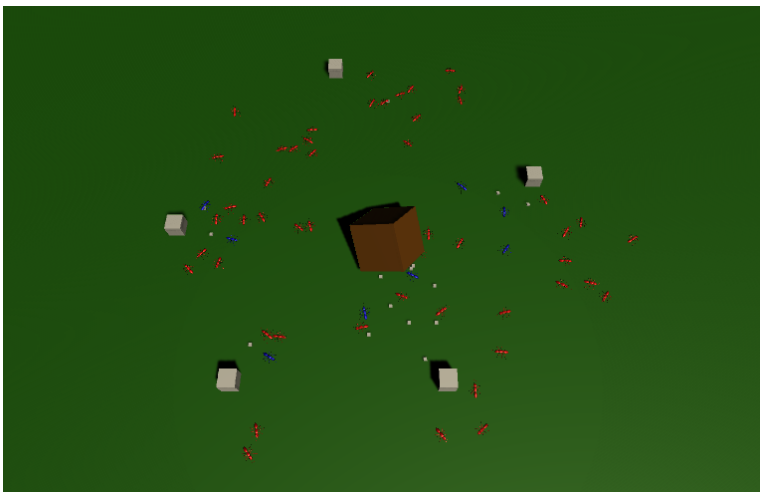
TODO

2 Mravenčí kolonie

TODO

2.1 Popis metody

TODO



Obrázek 1: Model mravenčí kolonie

3 Model Boid

Počítačový model pro řízení hejna zvířat jako jsou ptáci a ryby popsal Craig Reynolds v roce 1986. Jednotlivé jedince nazval Boidy. Samotný model pak prezentoval na konferenci SIGGRAPH o rok později.

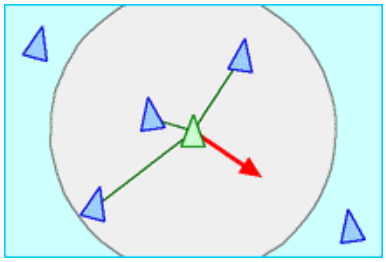
Od roku 1987 byl model Boid použit pro velké množství simulovaného chování hejna. Prvním, veřejně známým použitím byl film Batman se vrací (1992) od Tima Burtona. Obsahuje počítačově simulované chování hejna netopýrů a skupiny tučňáků, které bylo vytvořeno upravenou verzí modelu Boid. [3]

3.1 Popis modelu

Boid pracuje s agenty reprezentovanými pozicí a rychlostním vektorem. Model lze aplikovat pro pohyb ve dvou i třech rozměrech.

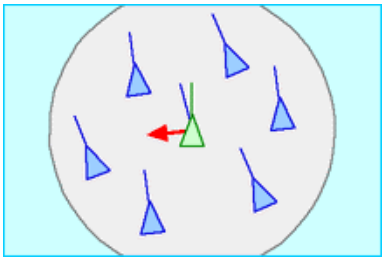
Každý agent se řídí třemi základními pravidly, podle kterých upraví svůj rychlostní vektor každý časový úsek. Jedná se o Separaci, Vyrovnání a Kohezi.

1. **Separace,**
   Agent se snaží nesrazit s jinými agenty. Upravuje proto svůj směrový vektor, aby se dostal od příliš blízkých agentů.



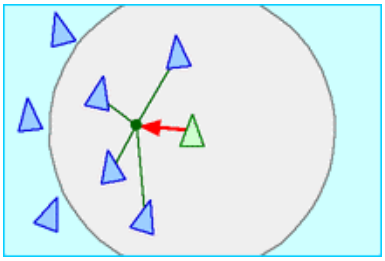
Obrázek 2: Separace (Separation) [3]

2. **Vyrovnání,**
   Agent se chce pohybovat stejným směrem a rychlostí jako jeho sousedé. Upravuje svůj vektor podle průměru z jeho okolí.



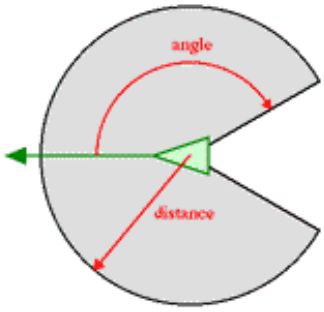
Obrázek 3: Vyrovnání (Alignment) [3]

3. **Koheze,**
   Agent upravuje svůj rychlostnní vektor tak, aby se dostal do pomyslného středu všech sousedních agentů.



Obrázek 4: Koheze (Cohesion) [3]

Každý boid má přímý přístup ke všem objektům ve scéně, nicméně tento model vyžaduje, aby reagoval pouze na své okolí. To je definováno maximální vzdáleností a zorným úhlem agenta. Můžeme přidávat i další pravidla jako třeba minimální vzdálenost nebo úhel mezi dvěmi agenty apod. Pravidla, podle kterého jedinec rozlišuje okolí, si můžeme představit jako smysly daného zvířete.



Obrázek 5: Okolí agenta [3]

Jak bylo na začátku kapitoly naznačeno, model můžeme různě zozšiřovat. Sám Reynolds přišel rok po představení svého modelu na stejné koferenci s rozšířením pro výhýbání se překážkám. Detailněji vše vysvětluje ve své práci Not bumping into things [4]. V podstatě se jedná o úpravu výsledného vektoru rychlosti pokud se před jedincem nachází nějaký objekt.

Pokud jednotlivým pravidlům přiřadíme váhy, můžeme následnými úpravami těchto vah měnit chování konkrétního jedince či celého hejna. Boidi tak můžou uzpůsobovat své chování v závislosti na prostředí. Například je-li v okolí dravec, budou mít mezi sebou větší rozestup, hlídat si větší okolí, rychleji reagovat na změnu směru atd.

4 Implementace

Implementace probíhala v jazyce *C#* a vývojovém prostředí Unity3D [5], ve kterém mám dlouholetou zkušenost.

4.1 Základní model Boid

Reynolds ve svém článku neuvádí jakým stylem jednotlivé pravidla řídit ani jaké váhy použít. Byla tedy vytvořena váha pro každé pravidlo, díky kterému můžeme řídit prioritu těchto pravidel.

Pro lepší vizuální představu jak model Boid funguje byl vytvořen jednoduchý editor, ve kterém je možné v reálném čase nastavit upravit váhy jednotlivým pravidlům a vidět změnu v chování hejna, aniž by bylo potřeba pouštět celou simulaci znovu. To umožnilo rychleji dosáhnout požadovaného chování hejna. Pro správné nastavení vah by teoreticky šel použít i evoluční algoritmus, avšak na tak jednoduchém příkladě není jasné jakým způsobem určovat fitness.

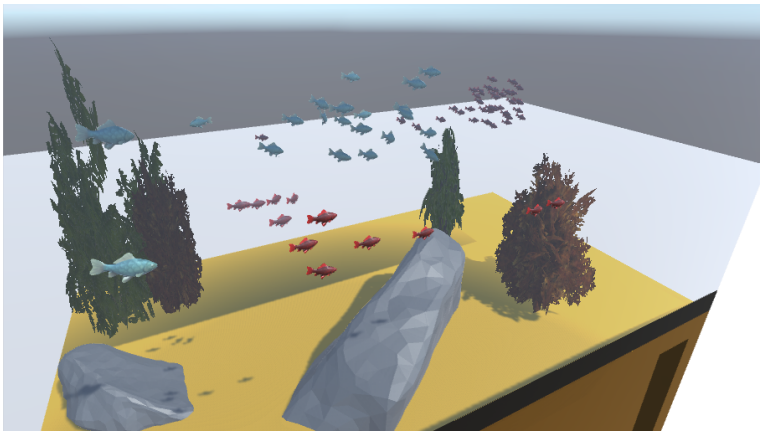
Podařilo se dosáhnout fungujícího modelu hejna, které však nekontrolovaně cestovalo po světě.

4.2 Rozšíření o pravidlo Cíl

Model Boid popisuje pouze chování jednotlivce v hejnu. Pro potřebu kontrolovat pohyb celého hejna, byl agent rozšířen o další pravidlo. Každý agent se snaží dostat do cíle, který je reprezentován pozicí bodu ve světě.

Od toho pravidla vyžadujeme, ať přitahuje agenta k cíli, ale neznehodnotí zbylé 3 pravidla s narůstající vzdáleností od cíle. Normalizování směrového vektoru k cíli se může zdát jako krok správným směrem. V praxi se však ukázalo, že když je takový agent velmi blízko svého cíle, normalizace vektor naopak zvětší. To má za následek příliš agresivní snahu jedince dostat se do cíle. Proto velikost směrového vektoru od jedince k cíli pouze ořízneme na hodnotu vzdálenosti, od které má být agent méně agresivní.

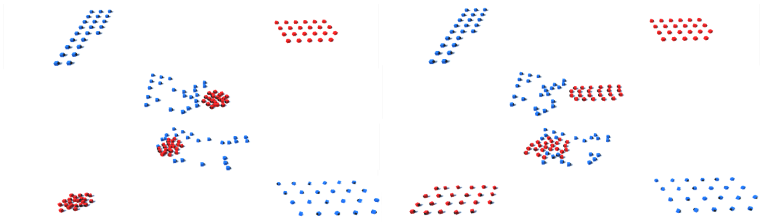
Pomocí nového pravidla je možné kontrolovat pozici hejna a pozicovat ho dle svých potřeb. Pro názornou ukázku vznikla scéna s akváriem a rybičkami. V určitém časovém intervalu nastavuji všem agentům (rybám) náhodně vygenerovaný cíl uvnitř akvária.



Obrázek 6: Hejno 20 modrých a 50 červených ryb

Na obrázku 4.2 můžeme vidět 2 různě početné a nastavené skupiny agentů. První je skupina 20 velkých modrých ryb, které jsou pomalé, drží se dál od sebe (velká váha u separace) a pomalu mění svůj směr. Druhá skupina obsahuje 50 malých červených ryb, které jsou rychlejší, agilnější, ale více se drží pohromadě (velká váha u koheze a vyrovnání).

Stejnými pravidly se dá simulovat i dav lidí, kteří se potřebují dostat z jednoho místa na druhé. Pro tyto účely byl každému jedinci nastaven odlišný cíl. Obrázek 7 znázorňuje srovnání chování 2 skupin poslaných proti sobě. Červená skupina na levém obrázku má nastavené váhy pro Kohezi a Vyrovnání převyšující váhu pravidla pro Cíl. Na pravém obrázku mají nastaveny váhy pro Kohezi a Vyrovnání tak, aby součet těchto vah nepřekročil váhu pro Cíl. Modrá skupina má nastavené pouze pravidlo Separace a Cíl. Pravidla pro Kohezi a Vyrovnání mají nulovou váhu. Každý agent má za úkol dostat se na druhou stranu a zaujmout pozici v pomyslné mřížce. Obrázky ve směru dolů jsou stavem s narůstajícím časem simulace.

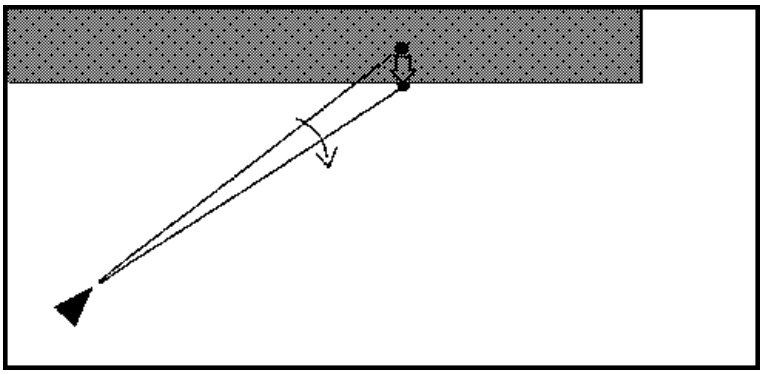


Obrázek 7: Srovnání vah v časech $t \in \{0, 20, 30, 70\}$

Z obrázku 7 můžeme usoudit, že poměr mezi váhou Cíle a váhami Koheze a Vyrovnání nemá na hejno jako celek vliv. V obou případech se hejno dostalo na druhou stranu. Avšak při malé váze pravidla Cíl nedokáže jedinec zaujmout svou koncovou pozici.

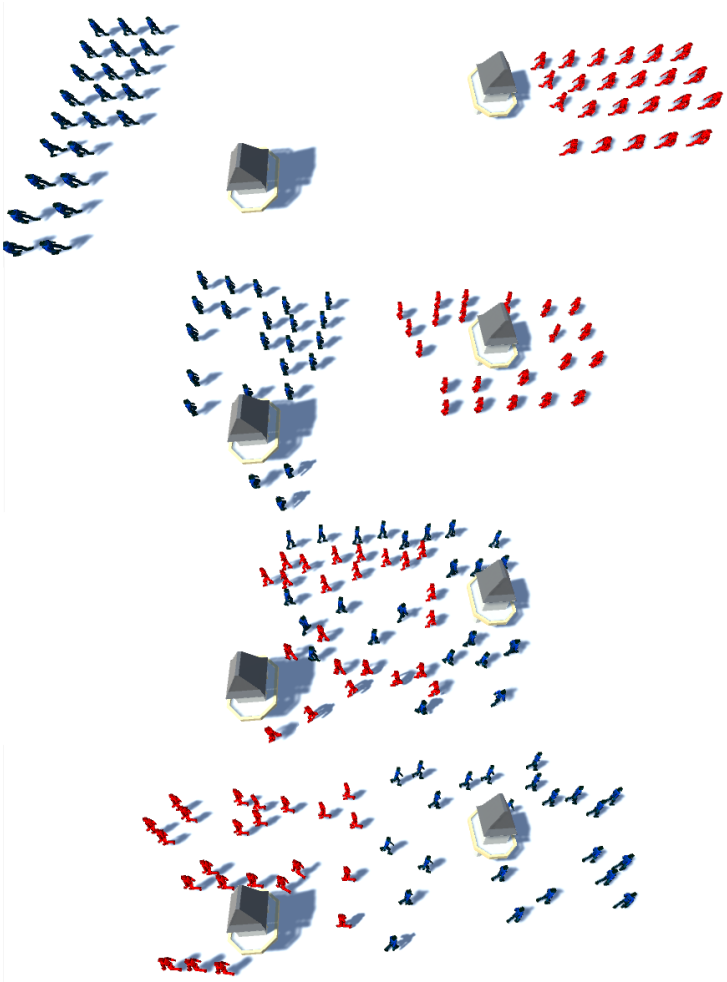
4.3 Rozšíření o pravidlo Vyhni se překážkám

Dalším přidaným pravidlem, díky kterému bude chování hejna více připomínat realitu je uhýbání před překážkami. Každý agent si v závislosti na své rychlosti kontroluje prostor před sebou. Pokud detekuje překážku, vychýlí svůj vektor rychlosti tak, aby se pokusil překážce vyhnout.



Obrázek 8: Vychýlení vektoru rychlosti při detekci překážky [4]

Pro test byly do scény přidány 2 překážky a puštěna obvyklá simulace dvou skupin, která si mají vyměnit místa. Na obrázku 9 můžeme pozorovat, že hejno překážku obteče a má tendenci se zase spojit do jednoho shluku.



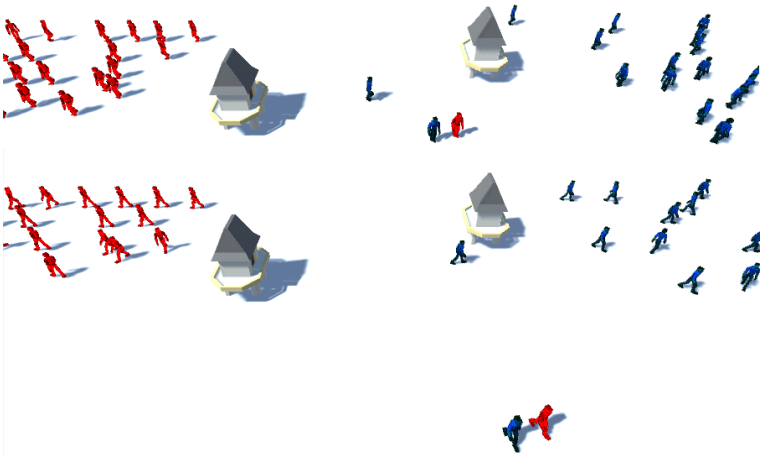
Obrázek 9: 2 skupiny agentů s detekcí překážek v časech $t \in \{0, 10, 20, 30\}$

5 Zhodnocení

TODO

5.1 Problémy

Přestože model Boid vykazuje známky složitého chování, stále se řídí jednoduhými pravidly, které mohou zapříčinit nechtěné odchylky. Jednou z nich je tzv. semknutí dvou agentů, kteří se navzájem přetlačují a jeden druhému nedokáží ustoupit. To má za následek nechtěnné chování, které může zapříčinit, že tito agenti začnou procházet překážkami.



Obrázek 10: Srážka a semknutí dvou agentů v časech $t \in \{100, 115\}$

6 Závěr

TODO

Reference

[1] Crowd and Multi-agent Simulation. *Gamma: Geometric Algorithms for Modeling, Motion, and Animation* [online]. [cit. 2017-05-04]. Dostupné z: http://gamma.cs.unc.edu/research/crowds/

[2] Dracula Untold Trailer. In: *Youtube* [online]. [cit. 2017-05-03]. Dostupné z: $https://www.youtube.com/watch?v =_2 aWqecTTuE$

[3] REYNOLDS, Craig. Boids: Background and Update. In: *Reynolds Engineering & Design* [online]. 1995 [cit. 2017-05-03]. Dostupné z: http://www.red3d.com/cwr/boids/

[4] REYNOLDS, Craig. Not Bumping Into Things. In: *Reynolds Engineering & Design* [online]. 1988 [cit. 2017-05-03]. Dostupné z: http://www.red3d.com/cwr/nobump/nobump.html

[5] *Unity 3D* [online]. [cit. 2017-05-04]. Dostupné z: https://unity3d.com

Seznam obrázků