FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA
KYBERNETIKY

Bakalářská práce

Kooperativní řízení multiagentního systému se zaměřením na detekci a ošetření kolizí

Petr Kolář

PLZEŇ 2024

FAKULTA APLIKOVANÝCH VĚD
ZÁPADOČESKÉ UNIVERZITY
V PLZNI

KATEDRA
KYBERNETIKY

Bakalářská práce

Kooperativní řízení multiagentního systému se zaměřením na detekci a ošetření kolizí

Petr Kolář

Vedoucí práce
Ing. Karel Kubíček

PLZEŇ **2024**

Citace v seznamu literatury:
KOLÁŘ, Petr. *Kooperativní řízení multiagentního systému se zaměřením na detekci a ošetření kolizí*. Plzeň, 2024. Bakalářská práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra kybernetiky. Vedoucí práce Ing. Karel Kubíček.

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Petr KOLÁŘ**
Osobní číslo:	**A21B0381P**
Studijní program:	**B0714A150005 Kybernetika a řídicí technika**
Specializace:	**Automatické řízení a robotika**
Téma práce:	**Kooperativní řízení multiagentního systému se zaměřením na detekci a ošetření kolizí**
Zadávající katedra:	**Katedra kybernetiky**

Zásady pro vypracování

1. Seznamte se s pojmy grafové teorie a jejich vztahem k multiagentnímu řízení a k topologii sítě.
2. Seznamte se s problematikou kooperativního řízení multiagentních systémů se zaměřením na detekování a ošetření kolizí.
3. Vyberte si algoritmus pro detekci a ošetření kolizí a ten zpracujte.
4. Zpracujte několik příkladů pro různé typy formací a ty si zvolte. Na nich ukažte možnosti Vámi vybraného algoritmu.
5. Vyhodnoťte výsledky poskytnuté Vámi vybraným algoritmem a diskutujte vhodnost vytvořeného řešení.

Rozsah bakalářské práce: **30-40 stránek A4**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

[1] J. Alexander Fax, "Optimal and Cooperative Control of Vehicle Formation", Disertation thesis, 2002.
[2] J. Alexander Fax and Richard M. Murray, "Information Flow and Cooperative Control of Vehicle Formations", 2004.
[3] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: algorithms and theory," in *IEEE Transactions on Automatic Control*, vol. 51, no. 3, pp. 401-420, March 2006, doi: 10.1109/TAC.2005.864190.
[4] Mehran Mesbashi and Magnus Egerstedt, "Graph Theoretic Methods in Multiagent Networks", 2010.
[5] Bing Yan, "Distributed Formation Control for Multi-agent Systems and Its Applications", PhD thesis, 2022, School of Electrical and Electronic Engineering Faculty of Sciences, Engineering and Technology The University of Adelaide.
[6] Xiajing Li, "Decentralized Collision Avoidance of Multi-Agent Systems in 3-Dimensional Space", Master thesis, Delft University of Technology.
[7] Jindřich Wolf, "Řízení kolaborativníchmulti-agentních dynamických systémů", Master thesis, 2019.

Vedoucí bakalářské práce: **Ing. Karel Kubíček**
Výzkumný program 1

Datum zadání bakalářské práce: **17. října 2023**
Termín odevzdání bakalářské práce: **20. května 2024**

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Radová

Doc. Dr. Ing. Vlasta Radová
vedoucí katedry

V Plzni dne 17. října 2023

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného akademického titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Západočeská univerzita v Plzni má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Plzni dne 17. května 2024

. . . .

Petr Kolář

V textu jsou použity názvy produktů, technologií, služeb, aplikací, společností apod., které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

Abstrakt

Tato bakalářská práce se zabývá kooperativním řízením multiagentního systému, zejména úlohami detekce a ošetření kolizí mezi agenty a překážkami. Cílem práce je vysvětlit a demonstrovat metody pro řízení skupin autonomních agentů tak, aby bylo dosaženo koordinovaného pohybu s minimálním rizikem kolize. Práce se věnuje analýze existujících algoritmů hejnového chování a jejich adaptací pro potřeby prevence kolizí s překážkami. Dále zkoumá využití konceptů teorie grafů a jejich vliv na topologii a vztahy mezi agenty. Následně se práce zaměřuje na vysvětlení vybraného algoritmu včetně popisu informačního toku, typu možných překážek a analýzou stability. V rámci práce jsou vytvořeny a otestovány simulace demonstrující shlukování agentů v definovaném cíli a jejich reakce na výskyt překážek. U všech příkladů jsou diskutovány vlastnosti systému a důsledky funkcí využitých na řízení multiagentního systému.

Abstract

This bachelor thesis focuses on cooperative control of a multi-agent system especially on the tasks of detecting and treating collisions between agents and obstacles. The aim of the thesis is to explain and demonstrate methods for controlling groups of autonomous agents to achieve coordinated movement with minimal risk of collision. The thesis analyzes existing swarm behavior algorithms and their adaptations for the purpose of preventing collisions with obstacles. Furthermore, the thesis explores the use of graph theory concepts and their impact on topology and relationships between agents. Subsequently, the thesis focuses on the explanation of the selected algorithm, including a description of the information flow, the type of possible obstacles and stability analysis. Simulations demonstrating the flocking of agents in a defined target and their reactions to the occurrence of obstacles are developed and tested. For all examples, the system properties and implications of the functions used to control the multi-agent system are discussed.

Klíčová slova

agent • flocking • kolektivní chování • kooperativní řízení • Laplacián • multiagentní řízení • shlukování • teorie grafů • vyhýbání se překážce • zpětná vazba

Poděkování

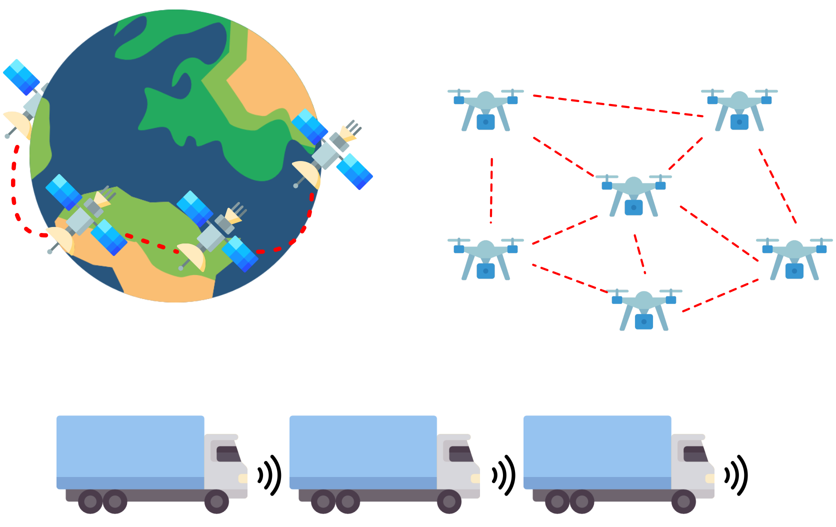
Tímto bych chtěl poděkovat svému vedoucímu práce Ing. Karlu Kubíčkovi za odbornou pomoc a cenné rady, bez nichž by práce nemohla být dokončena. Kdykoliv jsem potřeboval pomoc, vždy si našel čas a podporoval mě. Chtěl bych také poděkovat své rodině, která mě v době studia podporovala a vždy mi byla nápomocná. Nakonec bych chtěl poděkovat svým přátelům za ochotu a laskavost v době mého studia.

Obsah

Úvod 1

Kooperativní řízení multiagentních systémů je klíčovou doménou výzkumu v moderní automatizaci a robotice, která nabízí efektivní řešení pro složité úkoly vyžadující synchronizovanou spolupráci mezi mnoha agenty, jako jsou autonomní vozidla, průmyslové roboty či bezpilotní letouny.

Příkladem reálného využití vlastností multiagentního systému je projekt Starlink společnosti SpaceX, jenž představuje revoluční přístup k poskytování internetových služeb za využití konstelace satelitů na nízké oběžné dráze Země. Tato strategie umožňuje nabídnout nízkolatenční internet téměř po celém světě. Jednotlivé autonomní satelity musí efektivně komunikovat a koordinovat své pozice, aby nedošlo ke srážce a zároveň poskytovaly služby na požadované úrovni [1]. Dalším příkladem



Obrázek 1.1: Ukázka využití multiagentních systémů

může být tzv. *Truck Platooning* neboli technika pro řízení skupiny nákladních vozidel v těsné formaci s využitím technologií autonomního řízení pro synchronizaci rychlostí a brzdění. První vozidlo může být vybaveno speciálními aerodynamickými prvky, které snižují odpor vzduchu a umožňují následujícím vozidlům jet v zákrytu za ním. Tato technika může přispět k plynulosti provozu a rychlejší přepravě nákladů. Zároveň se snižuje spotřeba paliva a emisí, což přispívá ke zvýšení udržitelnosti lidské činnosti [2]. Systém může být rozšířen o drony, které dohlížejí na bezpečnost ze vzduchu a zároveň zajišťují včasné informace o překážkách, na které mohou vozidla včas reagovat a vyhnout se jim [3]. Na Obrázku (1.1) jsou znázorněny některé zde popsané příklady.

Jednou z důležitých vlastností kooperativního řízení je právě bezpečné vyhnutí se překážce a následný *flocking* neboli sloučení jednotlivých agentů do formace. Z tohoto důvodu je cílem této práce sjednotit informace o problematice multiagentních systémů, zejména v oblasti detekce a vyhýbání se překážkám a ověřit tyto znalosti simulacemi na specifických příkladech. V práci budou popsány všechny potřebné matematické pojmy, nutné k porozumění problému. Jedná se zejména o grafovou teorii a rovnice nutné ke konstrukci algoritmu. U jednotlivých simulací bude diskutováno, jak různé počáteční podmínky ovlivňují algoritmus.

Celá práce je rozdělena do několika hlavních kapitol, z nichž první kapitola má za úkol ukázat důvody a motivace, proč jsou v dnešní době multiagentní systémy žádané a zkoumané. Je zde také představena jejich historie a vysvětlen fenomén flocking, jež je nedílnou součástí řízení multiagentních systémů. Druhá kapitola se věnuje definování pojmu multiagentní systém. Dále uvádí základní pojmy grafové teorie, typy grafů a definuje tzv. alfa-mřížku. Třetí kapitola se zaměřuje na samotný algoritmus shlukování a řízení jednotlivých agentů. Jsou zde uvedeny jednotlivé funkce a rovnice, které jsou implementovány v algoritmu. Dále obsahuje popis jednotlivých typů agentů a analýzu stability. Čtvrtá kapitola upravuje stávající a definuje nové parametry nutné k rozšíření algoritmu o detekci a vyhnutí se překážkám. Dále popisuje typy překážek, které je schopen algoritmus zpracovat, a informačního tok mezi agenty. Poslední kapitola obsahuje jednotlivé typy simulací, lišící se různými počátečními podmínkami a typy překážek. Ty jsou porovnávány a je diskutován jejich vliv na fungování algoritmu. V závěru práce jsou zmíněny možná budoucí rozšíření práce a další možný vývoj v oblasti multiagentních systémů.

1.1 Motivace

Multiagentní systémy jsou v dnešní době hojně řešené téma, které s rozvojem výpočetní techniky a komunikace umožňuje jejich stále častější nasazení v praxi [4]. Jejich uplatnění lze nalézt zejména v oborech jako je robotika, energetika, řídící systémy a komunikační sítě.

Podíváme-li se do historie, myšlenka multiagentních systému se v literatuře objevuje už několik desítek let. Jedna z prvních zmínek o konceptu *agenta* se objevuje v práci J. V. Neumanna a O. Morgensterna, kteří tento termín použili ve své práci o herní teorii, publikované v padesátých letech minulého století [5]. Herní teorie poskytuje formální rámec pro analýzu strategických interakcí mezi racionálními účastníky, nazývanými agenty, v různých situacích rozhodování, kde jsou zahrnuty nejistoty a konkurence. Tato disciplína dále umožňuje modelování a vyhodnocování chování hráčů a optimalizaci jejich strategických rozhodnutí s cílem dosáhnout požadovaných výsledků, tedy výhry. Později se tento pojem *agenta* začal rozšiřovat. Principem autonomního agenta se zabýval R. Brooks, pracovník laboratoří MIT a princip inteligentních agentů pak popsal M. J. Wooldridge [6]. Mezi klíčové postavy, jež můžeme bezesporu zařadit mezi zakladatele oblastí kooperativní řízení a multiagentní systémy, patří J. A. Fax a Richard M. Murray [7] [8]. Jejich výzkum v těchto disciplínách přinesl významné poznatky, které se staly základem pro další pokrok v dané problematice. Jejich práce je široce respektována a často citována v nových vědeckých publikacích.

Jak již bylo výše zmíněno, koncept multiagentních systémů je využíván především ve spojení s distribuovanými systémy. Jejich implementace také reflektuje současnou tendenci přechodu z centralizovaného směrem k decentralizovanému řízení. Tento přístup k řízení nabízí výhody v podobě nižších provozních nákladů, efektivnějšího potlačení vnějších poruch a vyšší odolnosti proti chybám. Klíčovou charakteristikou decentralizovaného řízení je, že selhání jednoho agenta nemusí nutně vést k výrazným problémům v celém systému, pokud jsou zbylé entity schopny dosáhnout požadovaných cílů. Schopnost dosažení cílů se ovšem může lišit v závislosti na specifikách aplikace. Nicméně, decentralizace klade vyšší požadavky na komunikaci a programování konkrétní aplikace [9]. Naopak v centralizovaném systému může selhání nebo nedostupnost centrální řídící jednotky vést k paralýze celého systému, který pak není schopen plnit své původní funkce.

Z těchto důvodů se toto téma v dnešní době stává velmi diskutovaným v rámci široké škály výzkumů a projektů. Jedním z nich je výzkum v oblasti energetiky, kde se jedná o řešení termínu *Economic Dispatch Problem (EDP)* neboli decentralizovaného způsobu řízení energetických sítí. Cílem EDP je distribuovat požadovanou zátěž mezi X dostupných generátorů v síti za účelem minimalizace celkových nákladů na produkci energie na jednotku [10] [11]. Zůstaneme-li v oblasti energetiky, dalším zajímavým projektem je koncept *Smart City*. Ten představuje nový způsob uvažování o městském prostoru tím, že vytváří model, který integruje zdroje a systémy zelené energie, energetickou účinnost, udržitelnou mobilitu, ochranu životního prostředí a ekonomickou udržitelnost [12]. Cílem je využít technologie agentů k dosažení flexibilnějšího systému. V oblasti robotiky stojí za zmínku mezinárodní projekt RoboCup Soccer. Ten využívá vlastnosti multiagentních systémů k vytvoření týmu humanoidních robotů, kteří budou schopni porazit člověka ve světovém šampionátu do roku 2050 [13]. Mohli bychom odkázat na široký rozsah dalších autorů a jejich prací, avšak to nepředstavuje primární cíl této práce, a tak nám toto postačí, jako krátká motivace.

1.2 Flocking

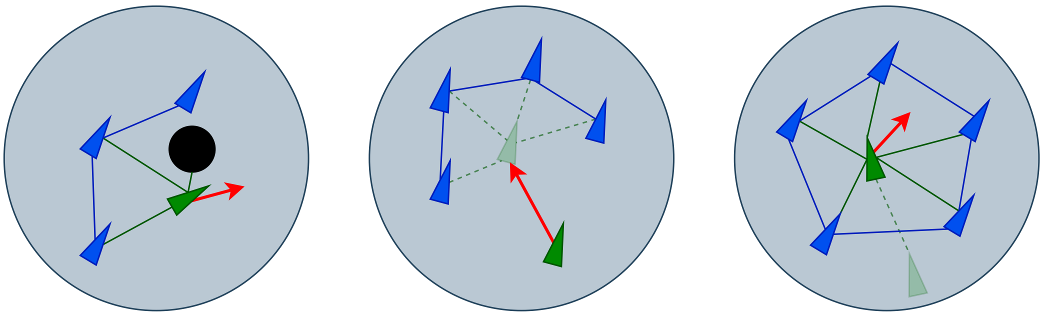
Flocking, jinými slovy shlukování nebo hejnové chování, je forma kolektivního chování velkého počtu interagujících agentů se společným skupinovým cílem. Přirozená synchronizace v přírodě sloužila jako významná inspirace pro zkoumání fenoménu shlukování. Výborným příkladem je kolektivní chování zvířecích skupin. Ať už jde o ptáky ve vzduchu, ryby ve vodě, nebo koně na louce, i když se každý jedinec pohybuje samostatně, skupina jako celek působí sjednoceně. Toto koordinované chování umožňuje zvířatům lepší úspěch při lovu, zvyšuje jejich schopnost detekovat hrozby a pomáhá jim šetřit energii během pohybu [14]. Technické využití hejnového chování zahrnuje široko-spektrální aplikace, jako je rozsáhlé distribuované detekování prostřednictvím mobilních senzorů a realizace vojenských úkolů, včetně průzkumu, dohledu a bojových akcí za pomoci skupin bezpilotních letounů (UAV). Shlukování také může být využito k automatickému programování internetových rádií s více kanály [15].

Flocking představil roku 1986 Craig Reynolds a uvedl tři heuristická pravidla, která využil ke tvorbě první simulace shlukování nazývané *Boids* [16]. Zde jsou pravidla, jež uvedl ve své práci:

1) Vyhnutí se překážce (separace)

2) Centrování shluku (soudržnost)

3) Sladění rychlosti (zarovnání)



Obrázek 1.2: Pravidla pro flocking, zleva separace, soudružnost a zarovnání

Na základě těchto pravidel, která jsou ilustrována na Obrázku (1.2), lze simulovat složité a realisticky působící shlukování pouhým využitím jednoduchých lokálních interakcí mezi jednotlivými agenty. Reynoldsovy práce tak otevřely cestu pro další výzkum v oblasti umělého života, robotiky a simulací, kde principy flockingu nacházejí uplatnění při modelování chování skupin.

Multiagentní systémy 2

Multiagentní systémy (MAS) jsou distribuované systémy, kde jednotlivé autonomní entity, známé jako agenti, mezi sebou interagují a spolupracují k dosažení společných nebo individuálních cílů. Tento druh systému využívá decentralizované rozhodování a komunikaci mezi agenty k řešení složitých úloh, které jsou pro jednotlivé agenty příliš náročné nebo neefektivní. Agenti MAS mohou být vybaveni schopnostmi jako je učení se, plánování nebo vnímání prostředí, díky čemu jsou schopni flexibilního reagování a adaptace na dynamické změny.

Pro správné definování, modelování a analýzu multiagentních systémů je nutné využít některé metody a nástroje z oblasti grafové teorie. Ta nabízí prostředky pro reprezentaci topologických struktur multiagentních systémů formou grafu, přičemž agenti jsou reprezentovány vrcholy a jejich komunikační či interakční vztahy jsou vyjádřeny hranami grafu. Důležitým pojmem v tomto kontextu je Laplaceův operátor, který umožňuje komplexní shrnutí vlastností daného systému do univerzální matice, reflektující komunikační dynamiku mezi agenty.

2.1 Teorie grafů

Teorie grafů je matematická disciplína zabývající se studií grafů, tedy struktur tvořených uzly, které jsou spojeny hranami. Graf G je uspořádaná dvojice (ν, ε), kde $\nu = \{v_1, v_2 ..., v_n\}$ je množina uzlů (vrcholů) a $\varepsilon = \{v_1, v_2; v_2, v_3; ...; v_n, v_m\}$ je množina hran. V tomto případě je každý uzel $v_n \in \nu$ brán jako agent a hrana $e_{nm} \in \varepsilon$ brána jako komunikační vazba mezi dvěma agenty. Hrany mohou být buď orientované, což znamená, že komunikace mezi agenty probíhá pouze v jednom směru, nebo neorientované, kdy komunikace probíhá obousměrně. Existují rovněž grafy, které obsahují ohodnocené hrany, kde každé spojení má přiřazenou určitou váhu. Ta většinou reprezentuje různé aspekty, jako jsou například náklady nebo cena. Grafy s ohodnocenými hranami jsou obvykle uváděny jako trojice $G = (\nu, \varepsilon, \omega)$, kde ω je množina hodnot jednotlivých hran grafu.

2.1.1 Základní pojmy

Nyní je potřeba uvést některé základní pojmy spjaté s grafovou teorií. Ty jsou zde rozděleny na pojmy grafové teorie a algebraické grafové teorie.

Většina informací obsažených v této a následujících kapitolách je čerpána z knihy *Introduction to Graph Theory* [17] napsané Robinem J. Wilsonem a knihy *Graph Theoretic Methods in Multiagent Networks* [18] od M. Mesbahiho a M. Egerstedta.

2.1.1.1 Pojmy grafové teorie

- **Stupeň uzlu** $deg(v_n)$ grafu G je počet hran grafu G, které uzel obsahují.
- **Řád grafu** G je celkový počet jeho uzlů, většinou je značen jako $|\mathcal{v}|$.
- **Velikost grafu** G je celkový počet jeho hran, většinou je značen jako $|\varepsilon|$.
- **Sled** v grafu G je taková posloupnost uzlů, kde každé dva po sobě jdoucí uzly mají mezi sebou hranu.
- **Tah** v grafu G je sled, v němž se neopakují hrany.
- **Cesta** v grafu G je sled, ve kterém se neopakují uzly a hrany.
- **Podgraf** *G1* může být nazýván podgrafem, pokud platí $\mathcal{v}(G1) \subset \mathcal{v}(G)$ a $\varepsilon(G1) \subset \varepsilon(G)$.
- **Smyčka** v grafu G je hrana, která spojuje uzel sám se sebou.
- **Přilehlost** v grafu G značí existenci hrany $e_{nm} = (v_n, v_m)$ mezi uzly v_n a v_m.

2.1.1.2 Pojmy algebraické grafové teorie

- **Čtvercová matice** — Matici $\boldsymbol{M}$ nazýváme čtvercovou, je-li počet řádků stejný jako počet sloupců, tedy $\boldsymbol{M} \in \mathbb{R}^{n \times n}$.
- **Symetrická matice** — Čtvercová matice $\boldsymbol{M}$ je symetrická právě tehdy, pokud platí $\boldsymbol{M} = \boldsymbol{M}^T$.
- **Definitnost matice** — Čtvercová matice $\boldsymbol{M}$ je pozitivně definitní, jestliže $x^T \boldsymbol{M} x > 0$ a negativně definitní, jestliže $x^T \boldsymbol{M} x < 0$ pro libovolný vektor $x \in \mathbb{R}^n \backslash \{0\}$.
- **Semidefinitnost matice** — Čtvercová matice $\boldsymbol{M}$ je pozitivně semidefinitní, pokud $x^T \boldsymbol{M} x \geq 0$ a negativně semidefinitní, pokud $x^T \boldsymbol{M} x \leq 0$ pro libovolný vektor $x \in \mathbb{R}^n$.

- **Vlastní čísla matice** — Vlastní čísla matice $\boldsymbol{M}$ jsou označovány jako λ_n a jsou výsledkem charakteristického polynomu $det(\lambda \boldsymbol{I} - \boldsymbol{M}) = 0$, kde $\boldsymbol{I}$ je jednotková matice.

- **Vlastní vektor matice** — Vlastní vektor v matice $\boldsymbol{M}$ je takový vektor, pro který platí $\boldsymbol{M}v = \lambda v$.

- **Nulový prostor matice** — Nulový prostor matice $\boldsymbol{M}$, většinou nazývaný jádrem matice neboli $ker(\boldsymbol{M})$, je takový prostor, jež je tvořen řešením homogenní soustavy lineárních rovnic x^T, tedy $\boldsymbol{M}x^T = o^T$.

- **Kroneckerův součin matic** — Kroneckerův součin matic, značen jako $\otimes$, je operace prováděna mezi dvěma maticemi, jejíž výsledkem je bloková matice neboli matice rozdělená do několika částí. Mějme matici $\boldsymbol{A}$ o rozměrech $n \times m$ a matici $\boldsymbol{B}$ o rozměrech $p \times q$, pak Kroneckerův součin $(\boldsymbol{A} \otimes \boldsymbol{B})$ těchto matic bude matice s rozměry $mp \times nq$ ve tvaru:

$$A \otimes B = (a_{nm} \cdot B) = \begin{bmatrix} a_{11}B & a_{12}B & a_{13}B & \cdots & a_{1m}B \\ a_{21}B & a_{22}B & a_{23}B & \cdots & a_{2m}B \\ a_{31}B & a_{32}B & a_{33}B & \cdots & a_{3m}B \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1}B & a_{n2}B & a_{n3}B & \cdots & a_{nm}B \end{bmatrix}.$$

  Zde je možné uvést numerický příklad na matici $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ a matici $B = \begin{bmatrix} 9 & 8 \\ 7 & 6 \\ 5 & 4 \end{bmatrix}$, jejichž výsledná matice Kroneckerova součinu bude mít rozměry 4×6 ve tvaru:

$$A \otimes B = \begin{bmatrix} 1 \cdot \begin{bmatrix} 9 & 8 \\ 7 & 6 \\ 5 & 4 \end{bmatrix} & 2 \cdot \begin{bmatrix} 9 & 8 \\ 7 & 6 \\ 5 & 4 \end{bmatrix} \\ 3 \cdot \begin{bmatrix} 9 & 8 \\ 7 & 6 \\ 5 & 4 \end{bmatrix} & 4 \cdot \begin{bmatrix} 9 & 8 \\ 7 & 6 \\ 5 & 4 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 9 & 8 & 18 & 16 \\ 7 & 6 & 14 & 12 \\ 5 & 4 & 10 & 8 \\ 27 & 24 & 36 & 32 \\ 21 & 18 & 28 & 24 \\ 15 & 12 & 20 & 16 \end{bmatrix}.$$

- **Stupňová matice** — Stupňová matice $\boldsymbol{D} \in \mathbb{R}^{n \times n}$, kde n je počet uzlů grafu a hodnoty i a j značí řádky a sloupce matice, je čtvercová diagonální matice, jejíž diagonálu tvoří stupně jednotlivých uzlů grafu a je definována takto:

$$D_{ij} = \begin{cases} \deg(v_i) & \text{pro } i = j, \\ 0 & \text{jinak.} \end{cases} \tag{2.1}$$

Příklad stupňové matice pro orientovaný a neorientovaný graf na Obrázku (2.1) vypadá takto:

$$D(O) = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad D(N) = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix}.$$

- **Matice sousednosti** — Matice sousednosti $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, kde n je počet uzlů grafu a hodnoty i a j značí řádky a sloupce matice, je čtvercová matice definovaná následovně:

$$A_{ij} = \begin{cases} 1 & \text{pro } (v_i, v_j) \in \varepsilon, \\ 0 & \text{jinak.} \end{cases} \tag{2.2}$$

Jednotlivé pozice (v_i, v_j) matice $\boldsymbol{A}$ značí vztahy mezi uzly grafu. Pokud mezi uzly existuje hrana, pak na tomto místě bude hodnota jedna, opačně hodnota nula. Pro neorientované grafy bude matice symetrická, avšak pro orientované grafy být symetrická nemusí, jelikož je nutné brát zřetel na směr hran grafu. Příklad matice sousednosti pro orientovaný a neorientovaný graf na Obrázku (2.1) vypadá takto:

$$A(O) = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \quad A(N) = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}.$$

- **Laplacián grafu** — Laplacián grafu je jedním z nejdůležitějších pojmů grafové teorie. Je to matice $\boldsymbol{L}$, která obsahuje důležité vlastnosti grafu, vyjádřená rozdílem stupňové matice $\boldsymbol{D}$ a matice sousednosti $\boldsymbol{A}$. Hlavní důležitou vlastností Laplaciánu je, že součet všech prvků v každém řádku je roven nule. Nejčastěji je Laplacián definován ve tvaru:

$$L = D - A. \tag{2.3}$$

Příklad Laplaciánu pro orientovaný a neorientovaný graf na Obrázku (2.1) vypadá takto:

$$L(O) = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ -1 & 0 & 2 & 0 & -1 \\ 0 & -1 & -1 & 2 & 0 \\ 0 & -1 & 0 & 0 & 1 \end{bmatrix}, \quad L(N) = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 3 & 0 & -1 & -1 \\ -1 & 0 & 3 & -1 & -1 \\ 0 & -1 & -1 & 3 & -1 \\ 0 & -1 & -1 & -1 & 3 \end{bmatrix}.$$

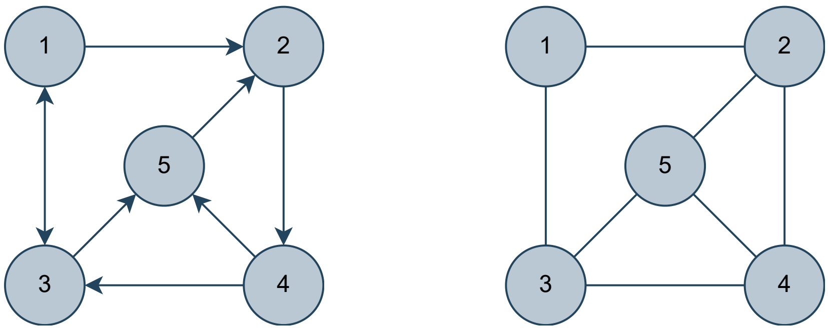
2.1.2 Typy grafů

Orientovaný graf

Orientovaný graf $\boldsymbol{G} = (\nu, \varepsilon)$, obdobně digraf, kde ν je množina vrcholů a ε množina hran $e_{ij} = (v_i, v_j)$, je takový graf, ve kterém mají hrany směr. Element v_i označíme jako $tail(v_i)$ hrany a v_j jako $head(v_j)$ hrany. Tyto elementy udávají směr hrany neboli $tail(v_i) \longrightarrow head(v_j)$. Dále je předpokládáno, že $tail(v_i) \neq head(v_j)$, což znamená, že graf neobsahuje vnitřní smyčky ze stejného vrcholu. Příkladem orientovaného grafu je Obrázek (2.1a).

Neorientovaný graf

Neorientovaný graf $\boldsymbol{G} = (\nu, \varepsilon)$, zobrazený na Obrázku (2.1b), kde ν je množina vrcholů a ε množina hran $e_{ij} = (v_i, v_j)$, je takový graf, ve které hrany směr nemají. Jedná se tedy o grafy, jež splňují $e_{ij} \in \varepsilon \Longrightarrow e_{ji} \in \varepsilon$.



(a) Příklad orientovaného grafu (b) Příklad neorientovaného grafu

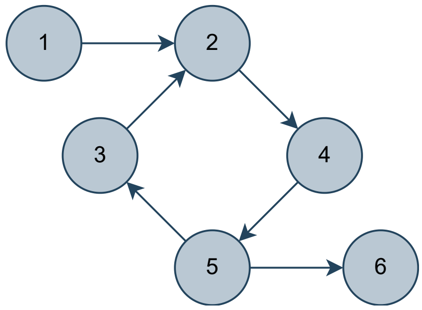
Obrázek 2.1: Příklad typů grafů s různou orientací hran

Cyklický graf

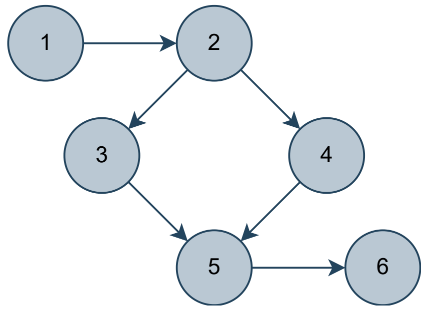
Cyklický graf, ilustrovaný na Obrázku (2.2a), je takový graf, který obsahuje cyklus (kružnici). Jinými slovy obsahuje uzavřenou posloupnost vrcholů, díky níž je možné se z uzlu v_i dostat zpět do stejného uzlu.

Acyklický graf

Acyklický graf, ilustrovaný na Obrázku (2.2b), je takový graf, ve kterém neexistuje cyklus. Není tedy možné se z uzlu v_i dostat zpět do stejného uzlu.



(a) Příklad grafu s cyklem



(b) Příklad grafu bez cyklu

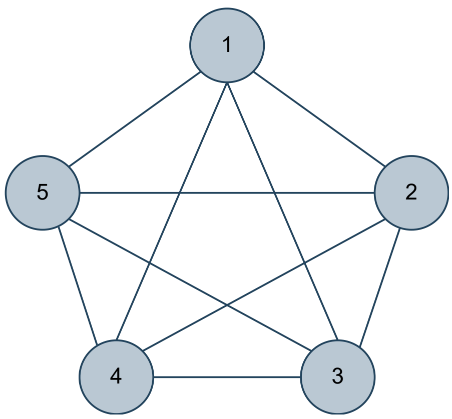
Obrázek 2.2: Příklad grafů s cyklem a bez cyklu

Periodický graf

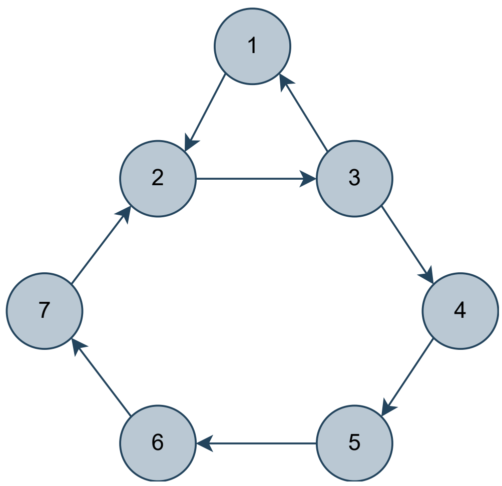
Periodický graf, zobrazený na Obrázku (2.3b), je takový graf, jehož množina délek všech jeho cyklů má společný dělitel $k \leq 1$.

Kompletní graf

Kompletní graf, zobrazený na Obrázku (2.3a), je takový graf, který obsahuje všechny možné hrany neboli každý uzel sousedí se všemi ostatními. Většinou je označován jako K_n, kde n je počet uzlů v grafu.



(a) Příklad kompletního grafu



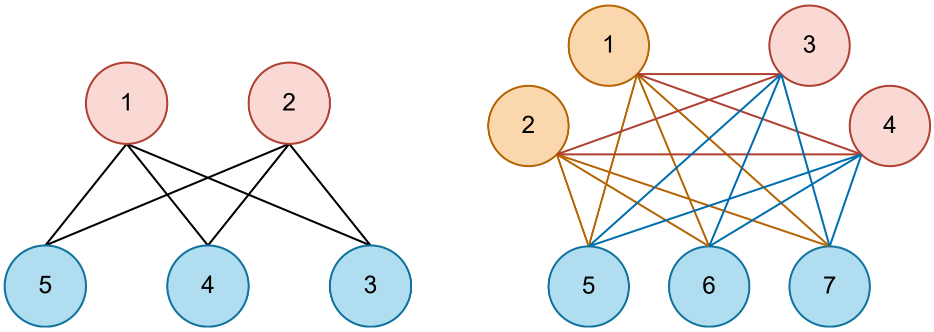
(b) Příklad periodického grafu

Obrázek 2.3: Příklad kompletního a periodického grafu

Bipartitní graf

Bipartitní graf $\boldsymbol{G} = (\nu, \varepsilon)$ je takový graf, jehož množinu uzlů lze rozdělit na dvě disjunktní množiny tak, že mezi žádnými dvěma uzly ze stejné množiny neexistuje hrana. Matematicky je to možné zapsat jako $\nu = \nu_1 \cup \nu_2, \nu_1 \cap \nu_2 = \emptyset; e = (u, v) \in \varepsilon, u \in \nu_1 \land v \in \nu_2$. Existují také K-partitní grafy, ve kterých jsou uzly rozděleny na

K disjunktních množin. Příklady těchto grafů jsou znázorněny na Obrázcích (2.4a) a (2.4b).



(a) Příklad bipartitního grafu (b) Příklad 3-partitního grafu

Obrázek 2.4: Příklad typů k-partitních grafů

2.1.3 Definice sousedů a Alfa-mřížek

Výše byly uvedeny základní pojmy z grafové teorie, které jsou nyní využity k definování nových pojmů, nutných pro vytvoření algoritmu.

Nejprve je nutné definovat množinu sousedů N jednotlivých uzlů (agentů). Pro graf $\boldsymbol{G} = (\nu, \varepsilon)$ bude množina sousedů:

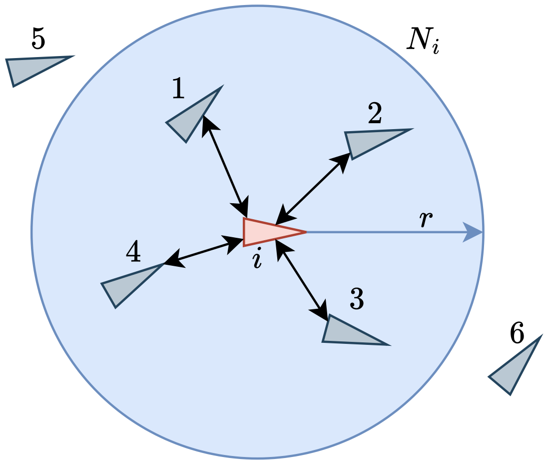
$$N_i = \{j \in \nu : a_{i,j} \neq 0\} = \{j \in \nu : (i,j) \in \varepsilon\}, \tag{2.4}$$

kde ν je množina uzlů, ε je množina hran a $a_{i,j}$ jsou prvky matice sousednosti. V celé práci je předpokládáno, že prvky $a_{ii} = 0$ pro všechna i neboli $(i,i) \notin \varepsilon$. Dále je vytvořen element $q_i \in \mathbb{R}^n$, kde $i \in \nu$, který označuje pozici uzlu v grafu. Vektor $q = (q_1, q_2, ..., q_i)$, tvořený těmito elementy, bude nazýván *konfigurací* všech uzlů grafu. Pár $(\boldsymbol{G}, q)$, který se skládá z grafu a konfigurace, pak bude nazýván *strukturou*.

Dále je označena konstanta $r > 0$ jako *interakční vzdálenost* mezi dvěma agenty. Pokud je vytvořena kolem agenta i kružnice o poloměru r, což může být vidět na Obrázku (2.5), potom všichni ostatní agenti, nacházející se také v této kružnici, jsou jeho sousedy, tedy jsou schopni s ním komunikovat. Množina těchto agentů je nazývána jako *množina přidružených sousedů* a definována jako:

$$N_i = \{j \in \nu : \|q_j - q_i\| < r\}, \tag{2.5}$$

kde $\|\cdot\|$ je Euklidovská vzdálenost. Z výše uvedeného je vidět, že graf $\boldsymbol{G}$ závisí na konfiguraci q. Graf $\boldsymbol{G}(q)$ je tedy označen jako *síť*. Tato síť může být digraf, pokud má každý agent specifickou interakční vzdálenost nebo neorientovaný graf, pokud jsou interakční vzdálenosti stejné. V této práci je uvažováno, že síť je orientovaný graf, na základě komunikačních vazeb mezi α-agenty.

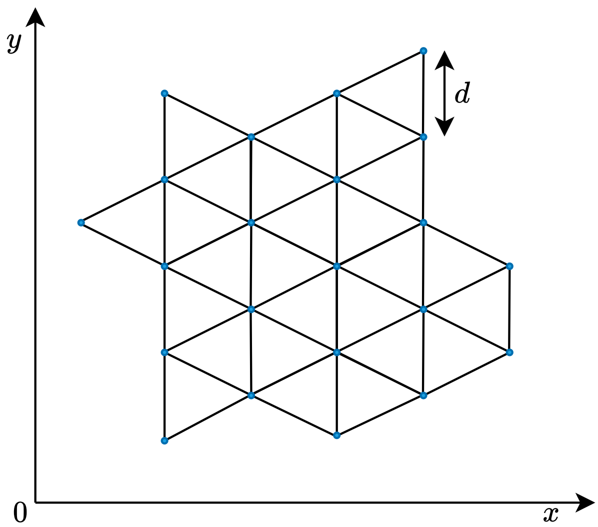


Obrázek 2.5: Agent a jeho množina přidružených sousedů

Jelikož je jedním z cílů této práce sloučení agentů do formace tak, aby mezi sebou udržovaly identické vzdálenosti, je potřeba vytvořit pravidlo pro síť $\boldsymbol{G}(q)$, díky kterému tento požadavek agenti splní. Toto pravidlo je nazváno *meziagentním algebraickým omezením* a definováno jako:

$$\|q_j - q_i\| = d, \forall j \in N_i(q) \tag{2.6}$$

Alfa-mřížkou bude nazývána konfigurace q, která splní omezení (2.6). Tato mřížka je znázorněna na Obrázku (2.6). Konstanta d značí měřítko a vztah $\varkappa = r/d$ vyjadřuje poměr alfa-mřížky. Název mřížky je odvozen od α-agenta, což je jeden z typů agentů, které jsou rozebírány dále v práci.



Obrázek 2.6: Příklad 2-D alfa-mřížky

Algoritmus pohybu a shlukování — 3

Tato kapitola se zabývá samotným algoritmem pro řízení pohybu a shlukování skupiny agentů. Nejprve jsou definovány typy agentů. První typ agenta se nazývá α-**agent** s pohybovou rovnicí $\ddot{q}_i = u_i$, kde u_i je zrychlení agenta a i definuje index i-tého agenta. Jedná se o fyzické agenty, jimiž jsou v reálném světě například ptáci, kteří jsou členy pohybující se skupiny a mají tendenci držet se ve vzdálenosti $d > 0$ od ostatních agentů. Toto je také důvod vzniku názvu alfa-mřížka. Dynamika této skupiny je pak vyjádřena pohybovou rovnicí:

$$\begin{cases} \dot{q}_i = p_i, \\ \dot{p}_i = u_i. \end{cases} \tag{3.1}$$

Druhý typ agenta je označen jako γ-**agent**, jež představuje cíl skupiny, jinými slovy bod, ke kterému iteruje skupina α-agentů. Tento cíl může být dynamický nebo statický, záleží na jeho definování. Dynamika tohoto agenta pak může být popsána rovnicí:

$$\begin{cases} \dot{q}_r = p_r, \\ \dot{p}_r = u_r, \end{cases} \tag{3.2}$$

kdy $(q_r(0), p_r(0)) = (q_d, p_d)$. Statický γ-agent pak bude mít fixní stav právě ve tvaru (q_d, p_d).

Před samotným návrhem algoritmu, je nutné definovat funkce, jež jsou využity pro jeho konstrukci. Výše uvedené a následující informace a značení o agentech, algoritmu a funkcích bylo převzato z Prací [19] [20], jejichž autorem je Reza Olfati-Saber.

3.1 Sigma norma

Sigma-norma ($\|z\|_\sigma$) je definována jako transformace vektoru dimenze m na nezáporné reálné číslo: $\mathbb{R}^m \longrightarrow \mathbb{R}_{\geq 0}$. Tato norma, na rozdíl od klasické normy vektoru, nemusí splňovat axiomy, které ji definují. Například nemusí být nula pro nulový vektor neboli je všude diferencovatelná. Sigma-norma je definována jako:

$$\|z\|_\sigma = \frac{1}{\epsilon}\left[\sqrt{1+\epsilon\|z\|^2}-1\right], \tag{3.3}$$

s parametrem $\epsilon > 0$. Dále je definován gradient této normy: $\sigma_\epsilon(z) = \nabla\|z\|_\sigma$, jako:

$$\sigma_\epsilon(z) = \frac{z}{\sqrt{1+\epsilon\|z\|^2}} = \frac{z}{1+\epsilon\|z\|_\sigma}. \tag{3.4}$$

Funkci $\|z\|_\sigma$ je využívána například ke konstrukci potenciálních funkcí pro řízení formace.

3.2 Nárazová funkce

Nárazová funkce $\rho(z)$ je skalární funkce, která plynule mění svoji hodnotu mezi 0 a 1. Je využívána pro konstrukci funkcí hladkého potenciálu s konečnými mezními hodnotami a hladkými maticemi sousednosti. Nárazová funkce je definována jako:

$$\rho_h(z) = \begin{cases} 1, & z \in \langle 0, h), \\ \frac{1}{2}\left[1+\cos\left(\pi\frac{(z-h)}{(1-h)}\right)\right], & z \in \langle h, 1\rangle, \\ 0 & \text{jinak,} \end{cases} \tag{3.5}$$

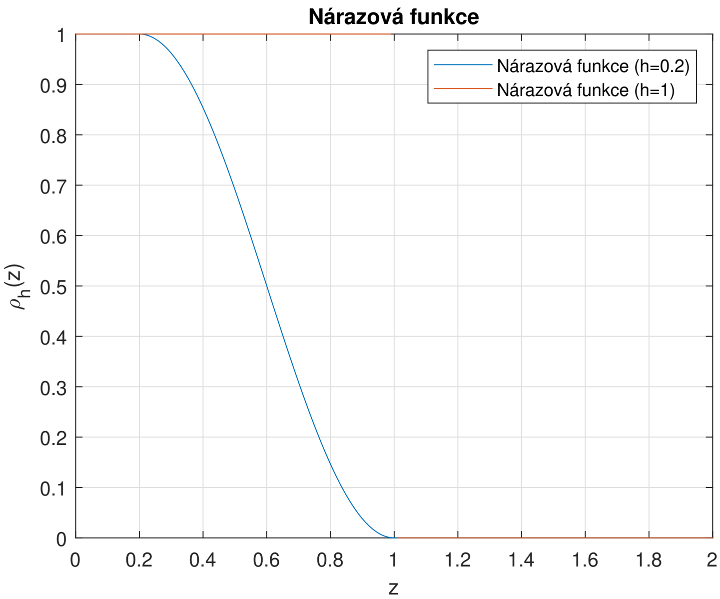
kde $h \in (0,1)$. Z definice je možné říci, že se jedná o hladkou funkci, jejíž derivace $\dot{\rho}_h(z) = 0$ na intervalu $\langle 1, \infty)$. Příklad průběhu nárazové funkce je na Obrázku (3.1).

3.2.1 Prostorová matice sousednosti

Pomocí Nárazové funkce (3.5) lze definovat prostorovou matici sousednosti $A(q)$, naplněnou elementy:

$$a_{ij}(q) = \rho_h\left(\|q_j - q_i\|_\sigma / r_\alpha\right) \in [0,1], \quad j \neq i, \tag{3.6}$$

kde $r_\alpha = \|r\|_\sigma$ a $a_{ii}(q) = 0$ pro všechna i a q. Pokud je $h = 1$, funkce $\rho_h(z)$ se stává indikátorovou funkcí, která je rovna 1 na intervalu $[0, 1]$ a 0 jinak. Použití indikátorové funkce vede na síť, která má závislé adjecentní prvky rovny 0 nebo 1.



Obrázek 3.1: Příklad průběhu nárazové funkce pro dva různé parametry

3.3 Kolektivní potenciál

Kolektivní potenciál skupiny agentů je nezáporná funkce $V(q)$, pro kterou platí $V : \mathbb{R}^{mn} \longrightarrow \mathbb{R}_{\geq 0}$. Dále platí, že jakékoliv řešení Algebraického omezení (2.6) je spojeno s lokálním minimem $V(q)$ a naopak. Funkce hladkého kolektivního potenciálu je definována jako:

$$V(q) = \frac{1}{2} \sum_i \sum_{j \neq i} \psi_\alpha(\|q_j - q_i\|_\sigma), \tag{3.7}$$

kde ψ_α je typ hladkého párového potenciálu. Párový potenciál je funkce, s jejíž pomocí je možné vyjádřit potenciální energii dvou objektů čistě na základě jejich vzdálenosti. V tomto případě je definována jako:

$$\psi_\alpha(z) = \int_{d_\alpha}^{z} \phi_\alpha(s)\, ds. \tag{3.8}$$

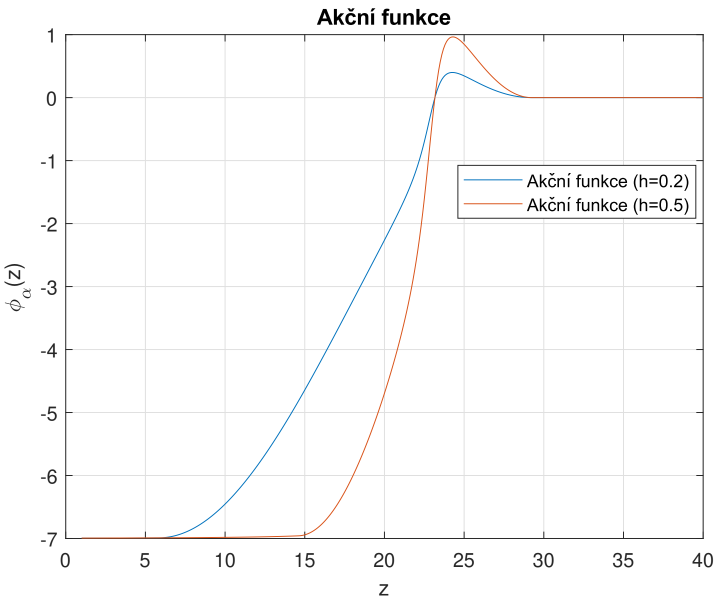
kde $d_\alpha = \|q_j - q_i\|_\sigma$ a $\phi_\alpha(s)$ je *Akční funkce* (3.9).

Běžný přístup pro vytvoření párového potenciálu s konečným omezením je přenásobení párového potenciálu Nárazovou funkcí (3.5). V tomto případě je však pro vytvoření koncového omezení využita akční funkce a následně její integrál pro definování Párového potenciálu (3.8). Tímto je zabráněno tomu, aby nárazová funkce byla negativní, a tím je získána konečná interakční vzdálenost $r > 0$ pro $\acute{\psi}_\alpha(z) = 0, \forall z \geq r$. Akční funkce, jejíž průběh je zobrazen na Obrázku (3.2), pak bude definována jako:

$$\phi_\alpha(z) = \rho_h(z/r_\alpha)\phi(z - d_\alpha), \tag{3.9}$$

$$\phi(z) = \frac{1}{2}\Big[(a+b)\sigma_1(z+c) + (a-b)\Big], \tag{3.10}$$

kde $\phi(z)$ je nesouměrná sigmoidální funkce s parametry, které splňují: $0 < a \leq b, c = |a-b|/\sqrt{4ab}$ pro zaručení $\phi(0) = 0$. Dále pak $\sigma_1 = z/\sqrt{1+z^2}$. Akční funkce je využívána pro konstrukci *gradietního termínu*, který bude popsán dále v práci.



Obrázek 3.2: Příklad průběhu akční funkce pro dva různé parametry

3.4 Návrh algoritmu

Základní struktura algoritmu pro pohyb a shlukování je založena na α-agentech, z nichž každý aplikuje řídící vstup, který je složen ze tří částí:

$$u_i = f_i^g + f_i^d + f_i^\gamma, \tag{3.11}$$

kde $f_i^g = -\nabla_{qi} V(q)$ je *gradientní termín*, f_i^d je *konsensuální termín* a f_i^γ je *navigační zpětná vazba*. Gradientní termín je používán v kontextu sil, které působí na agenta v závislosti na jeho pozici ve vztahu k ostatním agentům nebo prostředí. Jedná se o derivaci potenciální funkce, v našem případě Párového potenciálu (3.8), která reprezentuje *"energetickou krajinu"* systému. V kontextu flockingu působí gradientní termín tak, aby minimalizoval vzdálenost mezi agenty a zároveň zajistil, že nebudou příliš blízko sebe, což by mohlo vést ke kolizím.

Navigační zpětná vazba řídí celkový směr a chování skupiny, což znamená směřování k cíli nebo únik z určité oblasti. Tento termín je formulován jako funkce, která zahrnuje stav individuálních agentů a globální cíl, kterého se snaží dosáhnout.

Konsensuální termín zajišťuje, že se všichni agenti dohodnou na rychlosti a směru pohybu. V našem případě slouží také jako tlumící síla. Tento termín jsme však jako jediný ještě nedefinovali. Je založen na Laplaciánu (2.3), jehož matice bude mít vždy pravý vlastní vektor $1_n = (1, ..., 1)^T$ spojený s hodnotou vlastního čísla $\lambda_1 = 0$. Pro neorientovaný graf $\boldsymbol{G}$ s Prostorovou maticí sousednosti $\boldsymbol{A} = [A_{i,j}]$ (3.6) je předpokládán řídící vstup každého α-agenta jako:

$$u_i = \sum_{j \in N_i} a_{i,j}(p_j - p_i), \tag{3.12}$$

kde p_j a p_i jsou rychlosti α-agentů. Toto je jeden z typů protokolu konsensu. Jeho odvození a další typy je možné najít v Práci [21], jejíž autory jsou R. Olfati-Saber a R. M. Murray.

V tuto chvíli bylo uvedeno vše potřebné k definování algoritmu. Ten je značen jako $\boldsymbol{u_i = u_i^\alpha + u_i^\gamma}$, u_i^α je součet gradientního a konsensuálního termínu a u_i^γ je navigační zpětná vazba. Toto značení se hodí pro pozdější úpravy algoritmu. Dosazení do rovnice algoritmu tedy vypadá takto:

Algoritmus 1:

$$u_i = \sum_{j \in N_i} \phi_\alpha(\|q_j - q_i\|_\sigma)\boldsymbol{n}_{i,j} + \sum_{j \in N_i} a_{i,j}(q)(p_j - p_i) + f_i^\gamma(q_i, p_i), \tag{3.13}$$

$$u_i^\gamma := f_i^\gamma(q_i, p_i, q_r, p_r) = -c_1(q_i - q_r) - c_2(p_i - p_r), \quad c_1, c_2 > 0, \tag{3.14}$$

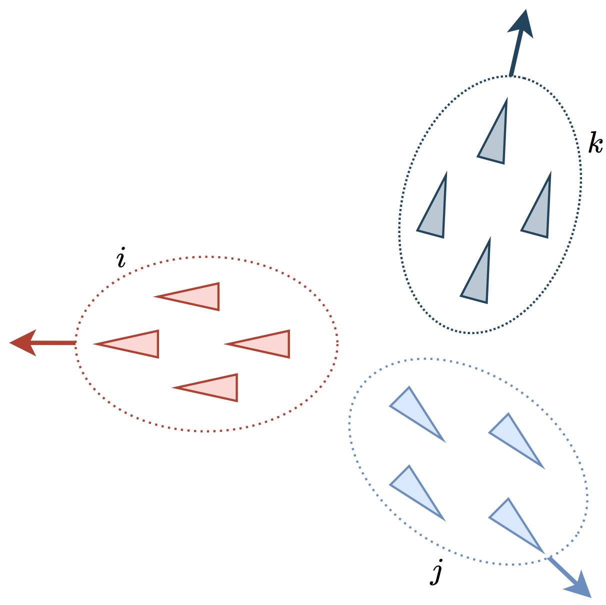
kde $\boldsymbol{n}_{i,j}$ je vektor podél přímky spojující polohy q_i a q_j a je vyjádřen jako:

$$n_{ij} = \sigma_\varepsilon(q_j - q_i) = \frac{q_j - q_i}{\sqrt{1 + \epsilon\|q_j - q_i\|^2}}, \tag{3.15}$$

a parametr $0 < \varepsilon < 1$ je fixní parametr Gradientu sigma normy (3.4). Ostatní parametry již v práci byly definovány. Dynamika skupiny α-agentů, popsanou Rovnicí (3.13), bude označena jako *kolektivní dynamika* tvořená Potenciální funkcí (3.7), Laplaciánem (2.3) a Navigační zpětnou vazbou (3.14) a bude zapsána následovně:

$$\begin{cases} \dot{q} = p, \\ \dot{p} = -\nabla V(q) - L(q)p + f_\gamma(q, p, q_r, p_r). \end{cases} \tag{3.16}$$

Zde by bylo dobré vyzdvihnout důležitost zpětné vazby. Pro malé množství agentů se striktními počátečními podmínkami by potřeba nebyla. Avšak v tomto případě, kdy je potřeba zajistit funkčnost pro počet agentů $n > 10$, by bez ní docházelo k *fenoménu fragmentace*, zobrazeného na Obrázku (3.3). Při jeho vzniku totiž α-agenti vytvoří několik malých skupin namísto jedné velké, což je velké úskalí pro flocking.



Obrázek 3.3: Ukázka fenoménu fragmentace

3.5 Analýza stability systému

V návaznosti na předchozí kapitolu, kde byla definována Dynamika skupiny α-agentů (3.16), je pro analýzu stability nutné tuto dynamiku rozdělit na *strukturální* a *translační* dynamiku. K tomu poslouží následující dekompoziční lemma:

Lemma 1 *Předpokládejme, že navigační zpětná vazba $f_\gamma(q,p)$ je lineární, tj. existuje rozklad $f_\gamma(q,p)$ v následujícím tvaru:*

$$f_\gamma(q,p,q_r,p_r) = g(x,v) + \mathbf{1}_n \otimes h(q_c,p_c,q_r,p_r), \tag{3.17}$$

kde dvojice (q_c,p_c) představuje polohu a rychlost středu shluku α-agentů. Poté lze kolektivní dynamiku skupiny α-agentů aplikující Algoritmus (3.13) rozložit jako i systémů druhého řádu (strukturální dynamika) následovně:

$$\begin{cases} \dot{x}_i = v, \\ \dot{v}_i = -\nabla V(x) - \hat{L}(x)v + g(x,v), \end{cases} \tag{3.18}$$

kde x_i a v_i je pozice a rychlost α-agenta vzhledem ke středu shluku dána takto:

$$\begin{cases} x_i = q_i - q_c, \\ v_i = p_i - p_c. \end{cases} \tag{3.19}$$

Dále dostaneme jeden systém druhého řádu (translační dynamika) jako:

$$\begin{cases} \dot{q}_c = p_c, \\ \dot{p}_c = h(q_c, p_c, q_r, p_r), \end{cases} \tag{3.20}$$

kde

$$g(x, v) = -c_1 x - c_2 x, \tag{3.21}$$

$$h(q_c, p_c, q_r, p_r) = -c_1(q_c - q_r) - c_2(p_c - p_r), \tag{3.22}$$

a (q_r, p_r) *je stav* γ*-agenta.*

Na základě tohoto lemma může být i samotná stabilita rozdělena na kombinaci dvou forem:

1. Stabilita určitých rovnovážných stavů strukturální dynamiky
2. Stabilita požadované hodnoty rovnovážného stavu translační dynamiky

V tomto případě je určení stability 2. části mnohem jednodušší, než pro 1.část. Pokud bude vycházeno z chování zvířat, translační dynamika nemusí mít nutně rovnovážný stav. Například hejno ryb může kroužit dokola v jedné oblasti nebo se pohybovat nepředvídatelným způsobem. Z technického hlediska se tedy nutně nezajímáme o rovnovážný stav, ale o možnost kontroly shluku (hejna) jako celku, což je zásadní pro řízení skupiny objektů z bodu A do bodu B. Bude tedy uvažována následující strukturální dynamika:

$$\Sigma : \begin{cases} \dot{x} = v, \\ \dot{v} = -\nabla U_\lambda(x) - D(x)v, \end{cases} \tag{3.23}$$

kde $U_\lambda(x)$ je funkce *agregátní potenciální funkce* definovaná jako:

$$U_\lambda(x) = V(x) + \lambda J(x). \tag{3.24}$$

Funkce $J(x) = \frac{1}{2}\sum_{i=1}^{n} \|x_i\|^2$ je *moment setrvačnosti* všech částic systému a $\lambda = c_1 > 0$ je parametr navigační zpětné vazby. Dále pak matice tlumení $D(x) = c_2 I_m + \hat{L}(x)$ je pozitivně definitní matice s $c_2 > 0$. Před uvedením analýzy stability je nutné definovat Hamiltonián systému, jenž je tvořen potenciální a kinetickou energií systému následovně:

$$H_\lambda(x, v) = U_\lambda(x) + K(v), \tag{3.25}$$

kde $K(v) = \frac{1}{2}\sum_i \|v_i\|^2$, což je funkce vyjadřující nesoulad rychlostí neboli kinetickou energii částic v systému. Následně je taktéž nutné definovat *soudružnost skupiny* a shluk.

Definice 1 *Uvažujme* $(q(\cdot), p(\cdot)) : t \to \mathbb{R}^{mn} \times \mathbb{R}^{mn}$ *jako stavovou trajektorii skupiny dynamických agentů za časový interval* $[t_0, t_f]$. *Skupina bude soudružná pro všechna* $t \in [t_0, t_f]$ *právě tehdy, když existuje kruh o poloměru* $R > 0$ *se středem v bodě* $q_c(t) = Ave\big(q(t)\big)$, *což je průměrná pozice všech agentů v daném čase, ve kterém jsou obsaženi všichni agenti po celý čas* $t \in [t_0, t_f]$, *tj.* $\exists R > 0 : \|x(t)\| \leq R, \forall t \in [t_0, t_f]$.

Definice 2 *Konfiguraci q množiny bodů v je nazývána shluk s interakční vzdáleností r pokud je síť* $G(q)$ *propojená. Dále skupinu* α*-agentů nazveme* dynamickým shlukem *na časovém intervalu* $t \in [t_0, t_f]$, *pokud jsou v každém okamžiku* $t \in [t_0, t_f]$ *shlukem.*

Výše uvedené definice jsou zásadní pro vysvětlení stability kolektivního chování skupiny α-agentů. Následující část vyšetřuje globální stabilitu Algoritmu (3.13). To je užitečné pro vytvoření shlukování pro generické sady počátečních podmínek.

Teorém 1 *Uvažujme skupinu* α*-agentů aplikující Algoritmus (3.13) s* $c_1, c_2 > 0$ *a strukturální dynamiku* $\sum$ *(3.23). Předpokládáme, že kinetická energie systému* $K\big(v(0)\big)$ *a moment setrvačnosti* $J\big(x(0)\big)$ *jsou konečné. Potom platí následující tvrzení:*

(i) *Skupina agentů zůstává soudružná pro všechna* $t \geq 0$.

(ii) *Většina řešení* $\sum$ *asymptoticky konverguje k rovnovážnému stavu* $(x_\lambda, 0)$, *kde* x_λ *je lokální minimum funkce* $U_\lambda(x)$.

(iii) *Rychlost všech agentů se asymptoticky shoduje.*

(iv) *Předpokládejme, že počáteční strukturální energie systému je menší než* $(k+1)c'$ *s* $c' = \psi_{\alpha(0)}$ *a* $k \in \mathbb{Z}_+$. *Pak je možné, že se srazí nanejvýš* ***k*** *různých párů* α*-agentů. Z tohoto důvodu musí být* $k = 0$.

Nejprve je blíže popsána *Část (i)*. Částicový systém se Strukturální dynamikou (3.23) a Hamiltoniánem (3.25) je striktně disipativní systém, což je systém, ve kterém se zvyšuje organizovanost, protože platí:

$$\hat{H}_\lambda(x, v) = -v^T\big(c_2 I_m + \hat{L}(x)\big)v = -c_2(v^T v) - v^T \hat{L}(x) v < 0, \quad \forall v \neq 0. \tag{3.26}$$

Strukturální energie $H_\lambda(x, v)$ tedy monotónně klesá pro všechna (x, v) a

$$H_\lambda\big(x(t), v(t)\big) \leq H_0 := H_\lambda\big(x(0), v(0)\big) < \infty. \tag{3.27}$$

Konečnost $H_0 = V\big(x(0)\big) + \lambda J\big(x(0)\big) + K\big(v(0)\big)$ vyplývá z předpokladu, že kolektivní potenciál $V(x)$, moment setrvačnosti $J(x)$ a kinetická energie $K(v)$ jsou zpočátku konečné. Pro všechna $t \geq 0$ pak platí:

$$U_\lambda\big(x(t)\big) \leq H_0, \quad K\big(v(t)\big) \leq H_0. \tag{3.28}$$

Zároveň ale $U\lambda(x) = V(x) + \frac{\lambda}{2}x^T x$, kde $\lambda > 0$ a $V(x) \geq 0$ pro všechna x, proto:

$$x^T(t)x(t) \leq \frac{2H_0}{\lambda}, \quad \forall t \geq 0. \tag{3.29}$$

To zaručuje soudružnost skupiny α-agentů pro všechna $t \geq 0$, jelikož pozice všech agentů se zdržuje v kruhu o poloměru $R = \sqrt{2H_0/\lambda}$ se středem v q_c. Tato kohezní vlastnost spolu s kinetickou energií neboli $K\big(v(t)\big) \leq H_0$ zaručuje ohraničenost řešení Strukturální dynamiky (3.23).

Část (ii) vyplývá z LaSalleho principu invariance, jenž je vysvětlen v práci K. Kubíčka a L. Bláhy [22]. Derivace $\dot{H}_\lambda(x, v) = 0$ implikuje $v = 0$. Na základě principu invariance je získán výsledek, že téměř každé řešení strukturální dynamiky asymptoticky konverguje k rovnovážnému stavu, jímž je $z_\lambda = (x_\lambda, 0)$, kde x_λ je lokální minimum agregátní potenciální funkce $U_\lambda(x)$.

Část (iii) vyplývá ze skutečnosti, že v asymptoticky zaniká. Z tohoto důvodu se tedy shodují i rychlosti všech agentů.

Pro vysvětlení *Části (iv)* je předpokládáno, že $H_0 < (k+1)c'$ a existuje více rozdílných párů agentů, které se srazí v daném čase $t_1 \geq 0$. Musí tedy existovat alespoň $k+1$ odlišných párů agentů, které se srazí v čase $t_1 \geq 0$. Z toho vyplývá, že kolektivní potenciál našeho částicového systému v čase $t = t_1$ je alespoň $(k+1)\psi_\alpha(0)$. Nicméně je získáno:

$$H_0 = V\big(x(0)\big) + \lambda J\big(x(0)\big) + K\big(v(0)\big) \geq V\big(x(0)\big) \geq (k+1)\psi_a(0). \tag{3.30}$$

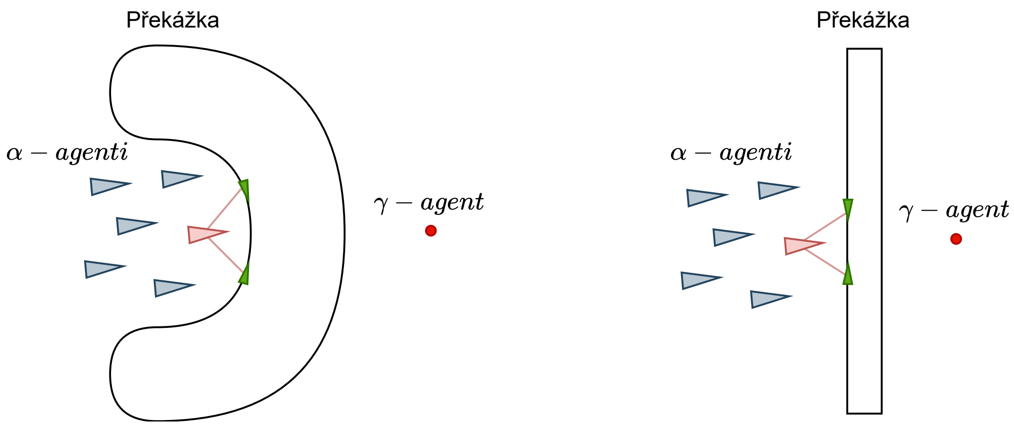
Tento výsledek je v rozporu s předpokladem, že $H_0 < (k+1)c'$. V žádném okamžiku $t \geq 0$ se tedy nemůže srazit více než k odlišných párů. Pak musí platit, že pro $k = 0$ se agenti nikdy nesrazí. Celý detailní postup, odvození a důkazy lze najít v Práci [19].

Rozšíření algoritmu o detekci a ošetření kolizí

4

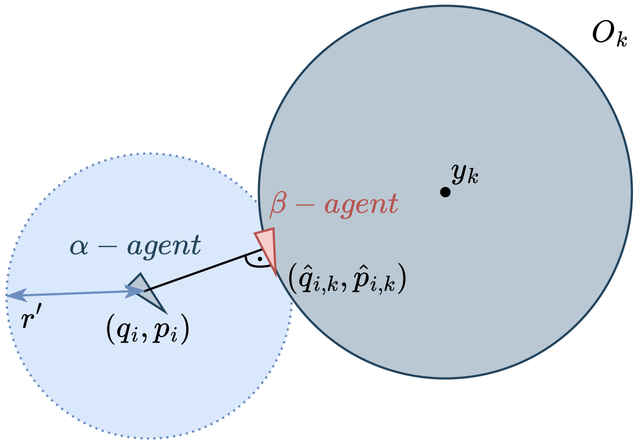
V předchozí kapitole byl popsán a vysvětlen základní algoritmus. Ten umožňuje pohyb skupiny agentů a dosažení cíle, což jsou základní podmínky pro to, aby mohl být rozšířen o detekci a ošetření kolizí. V této kapitole bude algoritmus rozšířen o schopnost reakce α-agentů na překážky, přesněji schopnost se jim vyhnout. Hlavní myšlenkou je vytvořit reprezentaci překážek pomocí nového typu agentů. Tento nový typ je nazýván β-**agent**. Jedná se o kinematického agenta, který je indukovaný α-agentem, jenž se nachází v blízkosti překážky.

Nejprve je potřeba definovat tvary překážek tak, aby se s nimi algoritmus dokázal vypořádat. Nejspíše nebude těžké si představit, že pro nekonvexní (konkávní) tvary se algoritmus dostane do problémů. Každý α-agent totiž bude indukovat více β-agentů najednou, čímž dojde ke konfliktu mezi úkolem vyhnutí se překážce a dosažení cíle (γ-agenta). Příkladem tohoto scénáře je Obrázek (4.1), jež znázorňuje tento problém. Daný α-agent indukuje dva β-agenty, a tím není schopen vytvořit jasné rozhodnutí o svém dalším pohybu.



Obrázek 4.1: Příklad nekonvexní a konvexní překážky, pro které algoritmus nelze použít

Stejná událost může nastat i pro specifické typy konvexních překážek. Tento problém je ilustrován na Obrázku (4.1). V tomto případě, pokud α-agenti projdou stěnou, nesplní přitom podmínku vyhnutí se překážce. Naopak pokud se jí pokusí vyhnou, zůstanou stát na místě, protože nastal stejný problém jako v předchozím příkladu. To vede na otázku proveditelnosti algoritmu, jelikož je potřeba se zamyslet nad typy překážek tak, aby ke konfliktu mezi úkoly nedocházelo.

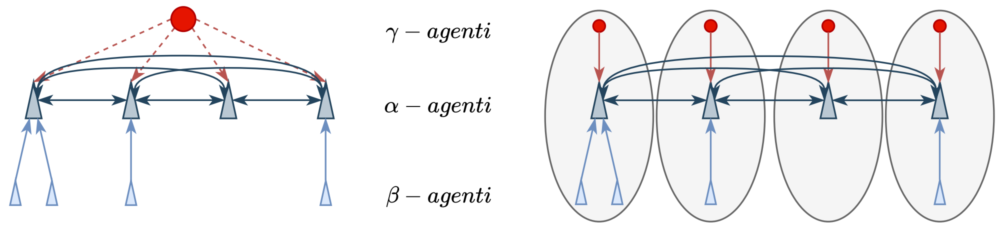


Obrázek 4.2: Parametry sférické překážky

Následující části práce se věnují pouze překážkám, jež jsou tvořené spojitými konvexními regiony v $\mathbb{R}^n$ s hladkými okraji a budou značeny O_k. Specificky se zabývají překážkami typu kruh, znázorněnými na Obrázku (4.2). Ten dále obsahuje znázornění indukce β-agenta a parametry nutné pro rozšíření algoritmu, bez nichž není možné splnit požadavek na vyhnutí se překážce. Tyto parametry jsou popisovány v dalších částech práce.

4.1 Informační tok systému

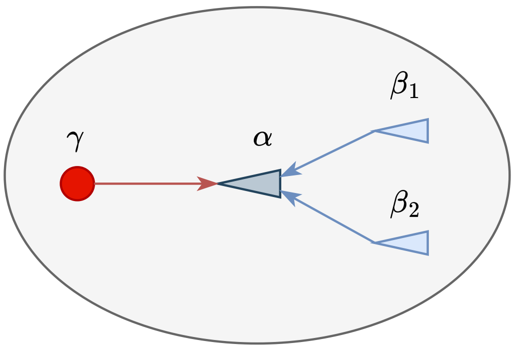
Než se pustíme do úpravy algoritmu, ukážeme zde princip výměny informací v celém systému tvořeným α-agenty, β-agenty a γ-agenty. Informační tok tohoto systému, který zahrnuje vyhýbání se překážkám a shlukování, má přirozenou hierarchickou strukturu. γ-agent má roli *virtuálního vůdce* a má na starosti navigaci skupiny α-agentů k cíli. V důsledku toho může být hierarchie označována jako *architektura virtuálního vůdce/následovníka* a je znázorněna na Obrázku (4.3). Přerušovaná čára značí výměnu informací mezi γ-agentem a α-agentem pouze v čase $t = 0$, v jiném případě se algoritmus stane centralizovaným, což může vést k problémům spojeným s leaderem. Pokud leader ztratí spojení s jednotlivými α-agenty, systém se zhroutí. Architektura virtuálního vůdce/následovníka by neměla být zaměňována s architekturou vůdce/následovníka, ve které je vůdcem jeden z fyzických agentů (např. vozidlo v multiagentním systému nebo ryba v hejnu).



Obrázek 4.3: Ukázka rozdílu mezi hierarchickou a peer-to-peer architekturou

Jelikož by výpočty, potřebné pro implementaci virtuálních agentů, musely být prováděny na vestavěných počítačích, kdy každý z těchto počítačů představuje fyzického agenta, neposkytuje to realistický obraz výpočetní architektury pro implementaci algoritmu. Hierarchická architektura je však užitečná pro pochopení důležitosti γ-agenta, který má roli sjednocujícího cíle, spojuje všechny α-agenty a vytváří propojenou síť.

Model informačního toku algoritmu je tedy vytvořen tak, že každému α-agentovy přiřadíme jednoho γ-agenta. Nová architektura je nazvána jako *peer-to-peer* síť, zobrazená na Obrázku (4.3). Ta je tvořena skupinou *makro-agentů*, z nichž každý se skládá z α-agenta a jeho odpovídajících β-agentů a γ-agentů. Příklad jejich struktury je na Obrázku (4.4). Dynamika těchto makro-agentů je velmi strukturovaná a je rozdělena na veřejné a soukromé části. Když spolu makro-agenti v síti komunikují, vyměňují si pouze veřejnou část informací (dynamika α-agenta). Biologický důsledek proveditelnosti sledování migrace skupiny dynamických agentů pomocí peer-to-peer sítě je, že hejna zvířat také nepotřebují vůdce [23]. Tento fakt o chování zvířat je znám desítky let.



Obrázek 4.4: Ilustrace makro agenta

4.2 Definice nových parametrů

S definicí β-agenta přibývá několik nových parametrů a také je potřeba upravit některé stávající. Nejprve jsou upraveny množiny sousedů N_i (2.4). Nechť $\nu_\alpha = \{1, 2, ..., n\}$ a $\nu_\beta = \{1', 2', ..., l'\}$ označují množinu indexů α-agentů a β-agentů (překážek). Prvky množiny ν_β jsou značeny apostrofem, čímž je kladen důraz na to, aby bylo zajištěno $\nu_\alpha \cap \nu_\beta = \emptyset$. α-agent je nazýván *sousedem překážky* O_k, pokud se kružnice kolem α-agenta a překážky O_k, jež značí interakční vzdálenost těchto objektů, překrývají. Dále je dobré podotknout, že α-agent může být sousedem více překážek, například pokud jsou překážky velmi blízko u sebe. Stávající množina sousedů N_i je tedy rozšířena na množinu α-sousedů a β-sousedů i-tého α-agenta následovně:

$$N_i^\alpha = \left\{ j \in \nu_\alpha : \|q_j - q_i\| < r \right\}, \tag{4.1}$$

$$N_i^\beta = \left\{ k \in \nu_\beta : \|\hat{q}_{i,k} - q_i\| < r' \right\}, \tag{4.2}$$

kde $r, r' > 0$ jsou interakční vzdálenosti α-agenta se sousedícími α-agenty a β-agenty a $\hat{q}$ značí konfiguraci všech β-agentů. Síť $\boldsymbol{G}(q)$ pak bude přejmenována na síť:

$$\boldsymbol{G}_{\alpha,\beta}(q) = \boldsymbol{G}_\alpha(q) + \boldsymbol{G}_\beta(q), \tag{4.3}$$

kde $\boldsymbol{G}_\alpha(q) = \left(\nu_\alpha, \varepsilon_\alpha(q)\right)$ je síť tvořená konfigurací všech α-agentů a $\boldsymbol{G}_\beta(q) = \left(\nu_\beta, \varepsilon_\beta(q)\right)$ je orientovaný bipartitní graf tvořený konfigurací q a množinou překážek $O = \left\{O_k \ : \ k \in \nu_\beta\right\}$. Podmínka $\nu_\alpha \cap \nu_\beta = \emptyset$ zaručuje správnost definice bipartitního grafu $\boldsymbol{G}_\beta(q)$.

Jako další je upraveno Algebraické omezení (2.6). Obdobně jako u množiny sousedů je omezení rozděleno na *meziagentní* a omezení *agent-překážka* následovně:

$$\begin{cases} \|q_j - q_i\| = d, & \forall j \in N_i^\alpha, \\ \|\hat{q}_{i,k} - q_i\| = d', & \forall k \in N_i^\beta. \end{cases} \tag{4.4}$$

Mřížka, která splňuje Rovnici (4.4), je nazývána *vázanou mřížkou*. Ta je tvořená dvojicí (q, O), jež se skládá z α-mřížky q a množiny překážek O. Výpočet poměru vázané mřížky zůstává stejný, tedy $\varkappa = r/d$ a $\varkappa' = r'/d'$.

Následuje úprava Kolektivního potenciálu (3.7), ze kterého vychází Akční funkce (3.9). Aby bylo možné docílit správné funkčnosti gradientního termínu v přítomnosti překážky, je upraven kolektivní potenciál na *vícedruhový kolektivní potenciál* tvořený:

$$V(q) = c_1^\alpha V_\alpha(q) + c_1^\beta V_\beta(q) + c_1^\gamma V_\gamma(q), \tag{4.5}$$

kde $c_1^\alpha, c_1^\beta, c_1^\gamma$ jsou pozitivní konstanty a (α, α), (α, β), (α, γ) jsou interakční potenciály definované jako:

$$V_\alpha(q) = \sum_{i \in V_\alpha} \sum_{j \in V_\alpha \setminus \{i\}} \psi_\alpha(\|q_i - q_j\|_\sigma), \tag{4.6}$$

$$V_\beta(q) = \sum_{i \in V_\alpha} \sum_{k \in N_i^\beta} \psi_\beta(\|\hat{q}_{i,k} - q_i\|_\sigma), \tag{4.7}$$

$$V_\gamma(q) = \sum_{i \in V_\alpha} (\sqrt{1 + \|q_i - q_r\|^2} - 1). \tag{4.8}$$

Funkce $V_\gamma(q)$ je spjatá s navigací skupiny agentů k cíli, potenciály $V_\alpha(q)$ a $V_\beta(q)$ mají stejnou funkčnost, jako Potenciál (3.7). Párový potenciál ψ_α je již taktéž definovaný, a tak zbývá definovat ψ_β jako *odpudivý párový potenciál* ve tvaru:

$$\psi_\beta(z) = \int_{d_\beta}^{z} \phi_\beta(s)\, ds \geq 0, \tag{4.9}$$

kde $d_\beta = \|d'\|$ a ϕ_β je *odpuzující akční funkce*, která z párového potenciálu vychází. Její předpis je:

$$\phi_\beta(z) = \rho_h(z/d_\beta)(\sigma_1(z - d_\beta) - 1), \tag{4.10}$$

kde $\sigma_1 = z/\sqrt{1 + z^2}$. Nárazová funkce (3.5) bude stejná a není ji třeba upravovat.

Jako další je vytvořena nová prostorová matice sousednosti. Jelikož již existuje Matice sousednosti (3.6), která bude potřebná pro funkčnost výsledného algoritmu, bude z ní vycházeno. Bude definována obdobně, ale bude představovat sousednost mezi α-agenty a β-agenty. Bude označována jako $B(q)$ a naplněna elementy:

$$b_{i,k}(q) = \rho_h(\|\hat{q}_{i,k} - q_i\|_\sigma / d_\beta). \tag{4.11}$$

Tímto jsou všechny parametry z předchozího algoritmu rozšířeny o nové, s jejichž pomocí bude definován nový algoritmus. Stále však zbývá určit parametry $\hat{q}_{i,k}$ a $\hat{p}_{i,k}$, které představují dynamiku β-agenta. Pro překážku O_k a její sousedící α-agenty ve stavu (q_i, p_i) je pozice a rychlost β-agenta dána následujícím lemma:

Lemma 2 *Nechť $\hat{q}_{i,k}$ a $\hat{p}_{i,k}$ s $(i, k) \in \nu_\alpha \times \nu_\beta$ značí pozici a rychlost β-agenta, generovanou α-agentem ve stavu (q_i, p_i), na překážce O_k. Potom:*

(i) Pro překážku tvaru kruhu (koule) s poloměrem R_k a středem v bodě y_k je poloha a rychlost β-agenta dána jako:

$$\begin{cases} \hat{q}_{i,k} = \mu q_i + (1 - \mu) y_k, \\ \hat{p}_{i,k} = \mu P p_i, \end{cases} \tag{4.12}$$

kde $\mu = R_k / \|q_i - y_k\|$, $a_k = (q_i - y_k)) / \|q_i - y_k\|$ a $P = I - a_k a_k^T$ je projekční matice.

Důkaz platnosti tohoto lemma lze nalézt v Práci [19].

4.3 Finální algoritmus

V tuto chvíli existují všechny předpoklady pro představení algoritmu, který bude brát ohled na překážky a bude vycházet z algoritmu pohybu a shlukování, definovaného Rovnicí (3.13), jenž byl již výše navrhnut. Skládá se tedy ze tří částí:

$$u_i = u_i^\alpha + u_i^\beta + u_i^\gamma, \tag{4.13}$$

kde u_i^α a u_i^β jsou složeny z gradientního a konsensuálního termínu a u_i^γ je navigační zpětná vazba. Jednotlivé části také vyjadřují interakce mezi agenty a překážkou:

$u_i^\alpha \longrightarrow$ Interakce mezi agenty (α, α)

$u_i^\beta \longrightarrow$ Interakce mezi agentem a překážkou (α, β)

$u_i^\gamma \longrightarrow$ Distribuovaná zpětná vazba

Matematické rovnice popisující algoritmus, jsou tvořené výše definovanými parametry a definovány následovně:

Algoritmus 2:

$$u_i^\alpha = c_1^\alpha \sum_{j \in N_i^\alpha} \phi_\alpha(\|q_j - q_i\|_\sigma) n_{i,j} + c_2^\alpha \sum_{j \in N_i^\alpha} a_{i,j}(q)(p_j - p_i), \tag{4.14}$$

$$u_i^\beta = c_1^\beta \sum_{k \in N_i^\beta} \phi_\beta(\|\hat{q}_{i,k} - q_i\|_\sigma) \hat{n}_{i,k} + c_2^\beta \sum_{j \in N_i^\beta} b_{i,k}(q)(\hat{p}_{i,k} - p_i), \tag{4.15}$$

$$u_i^\gamma = -c_1^\gamma \sigma_1(q_i - q_r) - c_2^\gamma (p_i - p_r), \tag{4.16}$$

kde $\sigma_1 = z/\sqrt{1 + \|z\|^2}$ a c_η^ν jsou pozitivní konstanty pro $\eta = 1, 2$ a $\nu = \alpha, \beta, \gamma$. Hodnoty q_r a p_r značí pozici a rychlost cíle, ke kterému agenti iterují. Nakonec zbývá definovat vektory n_{ij} a $\hat{n}_{ik}$, jež jsou definovány pomocí Gradientu sigma normy (3.4) jako:

$$n_{ij} = \frac{q_j - q_i}{\sqrt{1 + \epsilon\|q_j - q_i\|^2}}, \quad \hat{n}_{ik} = \frac{\hat{q}_{i,k} - q_i}{\sqrt{1 + \epsilon\|\hat{q}_{i,k} - q_i\|^2}}, \tag{4.17}$$

kde $n_{i,j}$ je vektor podél přímky spojující polohy q_i a q_j a $\hat{n}_{i,k}$ je vektor podél přímky spojující polohy q_i a $\hat{q}_{i,k}$.

4.4 Výsledky simulací

V této sekci bude představeno několik simulací, na kterých jsou ukázány vlastnosti a funkčnost algoritmu. Simulace jsou prováděny ve 2-D pro různé typy formací a počty α-agentů a překážek. Příklad struktury algoritmu a zdrojové soubory jsou obsaženy v Příloze (A). Každá simulace obsahuje popis zkoumané funkčnosti, parametry, pro které byla simulace spuštěna, grafy průběhu simulace a její výsledky.

Zde jsou uvedeny parametry, jež jsou fixní v průběhu všech simulací: poměr alfa-mřížky $\varkappa = 1.2$, interakční vzdálenost α-agenta $r = \varkappa \cdot d$, měřítko alfa-mřížky $d = 17$, které je stejné téměř ve všech simulacích a o jeho případné změně je informováno. Dále interakční vzdálenost β-agenta $r' = 1.2 \cdot d'$, parametr σ-normy $\varepsilon = 0.1$, parametry funkce $\phi(z)$ $a = b = 5$, parametr nárazové funkce $h = 0.2$ pro $\phi_\alpha(z)$, nárazové funkce $h' = 0.7$ pro $\phi_\beta(z)$ a velikost krokování simulací je $d_t = 0.03$.

Jako další jsou zde zmíněny parametry, které jsou popsány u každé simulace zvlášť: počet agentů n, počáteční rozmístění α-agentů a překážek, umístění cíle (γ-agenta), měřítko vázané mřížky d' a parametry c_η^ν pro Rovnice (4.14), (4.15) a (4.16) výše uvedeného algoritmu, kde $\nu = \alpha, \beta, \gamma$ a $\eta = 1, 2$. Ty jsou velmi důležité, jelikož ovlivňují intenzitu působení jednotlivých složek algoritmu na α-agenty následujícím způsobem:

$$c_1^\alpha, c_1^\beta > 0 \quad \longrightarrow \quad \text{Intenzita působení gradientních termínů}$$

$$c_2^\alpha, c_2^\beta > 0 \quad \longrightarrow \quad \text{Intenzita působení konsensuálních termínů}$$

$$c_1^\gamma, c_2^\gamma > 0 \quad \longrightarrow \quad \text{Intenzita působení navigační zpětné vazby}$$

Experimentálně bylo zjištěno, že pro ideální funkčnost algoritmu, je nutné, aby $c_1^\gamma < c_1^\alpha < c_1^\beta$ a $c_2^\gamma = 2\sqrt{c_1^\gamma}$. Tímto je zajištěno, že se některá z výše uvedených složek algoritmu nestane dominantní, což by mohlo vést k destabilizaci a pádu algoritmu.

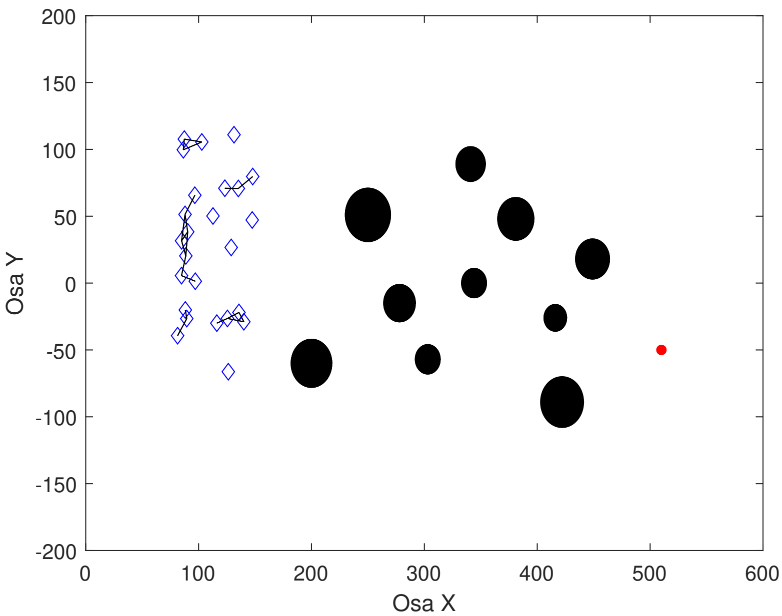
Nakonec před samotným simulováním uvedeme vysvětlení jednotlivých výsledků na Obrázcích (4.8, 4.12, 4.16, 4.20, 4.24). Ty obsahují digraf, jenž slouží k demonstraci tvaru výsledného shluku a komunikačních vazeb mezi α-agenty. Dále obsahují průběh trajektorie středu shluku a trajektorie jednotlivých agentů. Dalšími grafy jsou graf průběhu konektivity a graf průběhu rychlostí jednotlivých agentů. Konektivita značí vlastnost alfa-mřížky, že od kteréhokoliv α-agenta v alfa-mřížce existuje cesta ke všem ostatním, a je spočtena jako hodnost Matice sousednosti (3.6). Rychlosti jednotlivých α-agentů jsou spočteny jako $v_i = v_x + v_y$, kde v_x jsou rychlosti jednotlivých agentů ve směru osy x a v_y rychlosti ve směru osy y. S důvodu většího počtu α-agentů mohou být grafy rychlostí agentů nepřehledné, jelikož zobrazují výsledky pro všechny α-agenty. Grafy tedy slouží spíše k představě o chování algoritmu. Jako poslední je graf řízení jednotlivých typů agentů. Ten ukazuje vliv všech tří rovnic algoritmu na agenty ve směru osy x a osy y.

4.4.1 1. Simulace–Ukázka interakcí mezi α-agenty a β-agenty

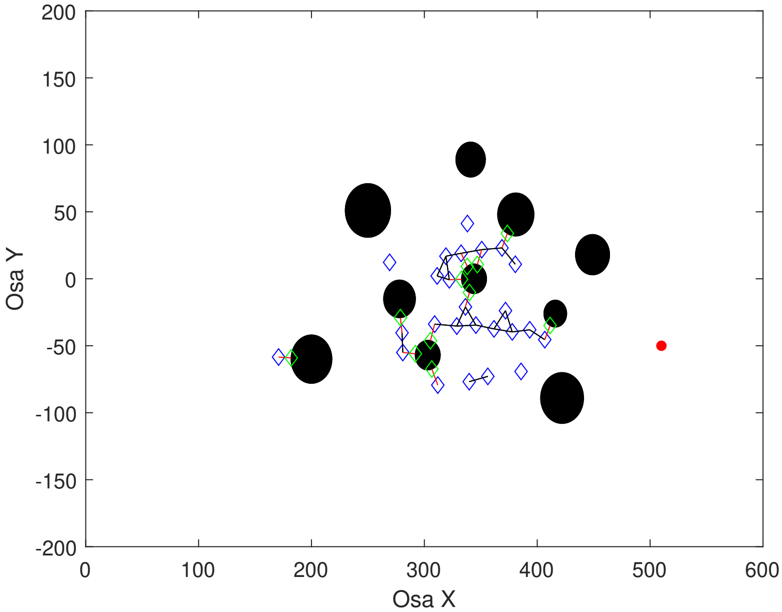
První simulace má za úkol znázornit interakce mezi α-agenty a β-agenty. α-agenti jsou znázorněni modrými body, β-agenti zelenými body, γ-agent červeným bodem a překážky jsou černé body, což je možné vidět na Obrázku (4.6). Dále pak vazby (α,α) jsou zobrazeny černými čárami a vazby (α,β) červenými čárami, což je taktéž vidět na Obrázku (4.6). Toto značení bude stejné u všech následujících simulací. Na Obrázku (4.5) je počáteční stav simulace. α-agenti byli náhodně generováni v rozmezí $(80, 150)$ na ose x a $(-80, 120)$ na ose y, γ-agent má souřadnice $[510, -50]$ a překážky byly náhodně vygenerovány do následující matice M:

$$M_1 = \begin{bmatrix} 422 & 381 & 449 & 200 & 278 & 344 & 250 & 416 & 341 & 303 \\ -89 & 48 & 18 & -60 & -15 & 0 & 51 & -26 & 89 & -57 \\ 19 & 16 & 15 & 18 & 14 & 11 & 20 & 10 & 13 & 11 \end{bmatrix}, \tag{4.18}$$

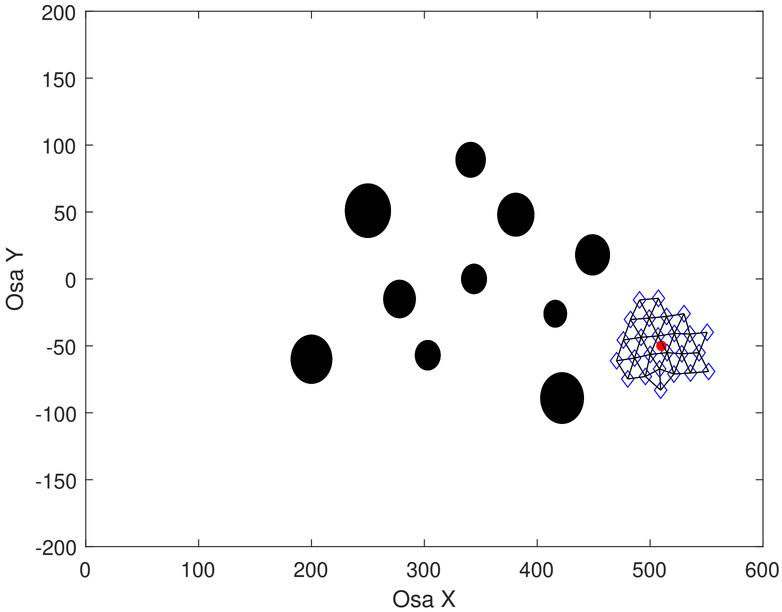
kde první řádek značí souřadnice na ose x, druhý souřadnice na ose y a třetí poloměry překážek. Tento formát bude opět následovat ve všech následujících simulacích. Simulace byla spuštěna pro parametry: $n = 25$, $d' = 0.7$, $c_1^\alpha = 38$, $c_1^\beta = 600$, $c_1^\gamma = 28$ a $c_2^\gamma = 2\sqrt{c_1^\gamma}$. Obrázek (4.6) ukazuje průběžný stav simulace, kde je vidět chování α-agentů v okolí překážek, indukování β-agentů a jejich interakce. Když se α-agenti dostanou k cíli skupiny, vytvoří shluk ve tvaru alfa-mřížky se středem shluku v bodě $[510, -50]$ neboli na místě γ-agenta, což ukazuje Obrázek (4.7). Výsledky na Obrázku (4.8) budou porovnány v diskusi výsledků s ostatními simulacemi.



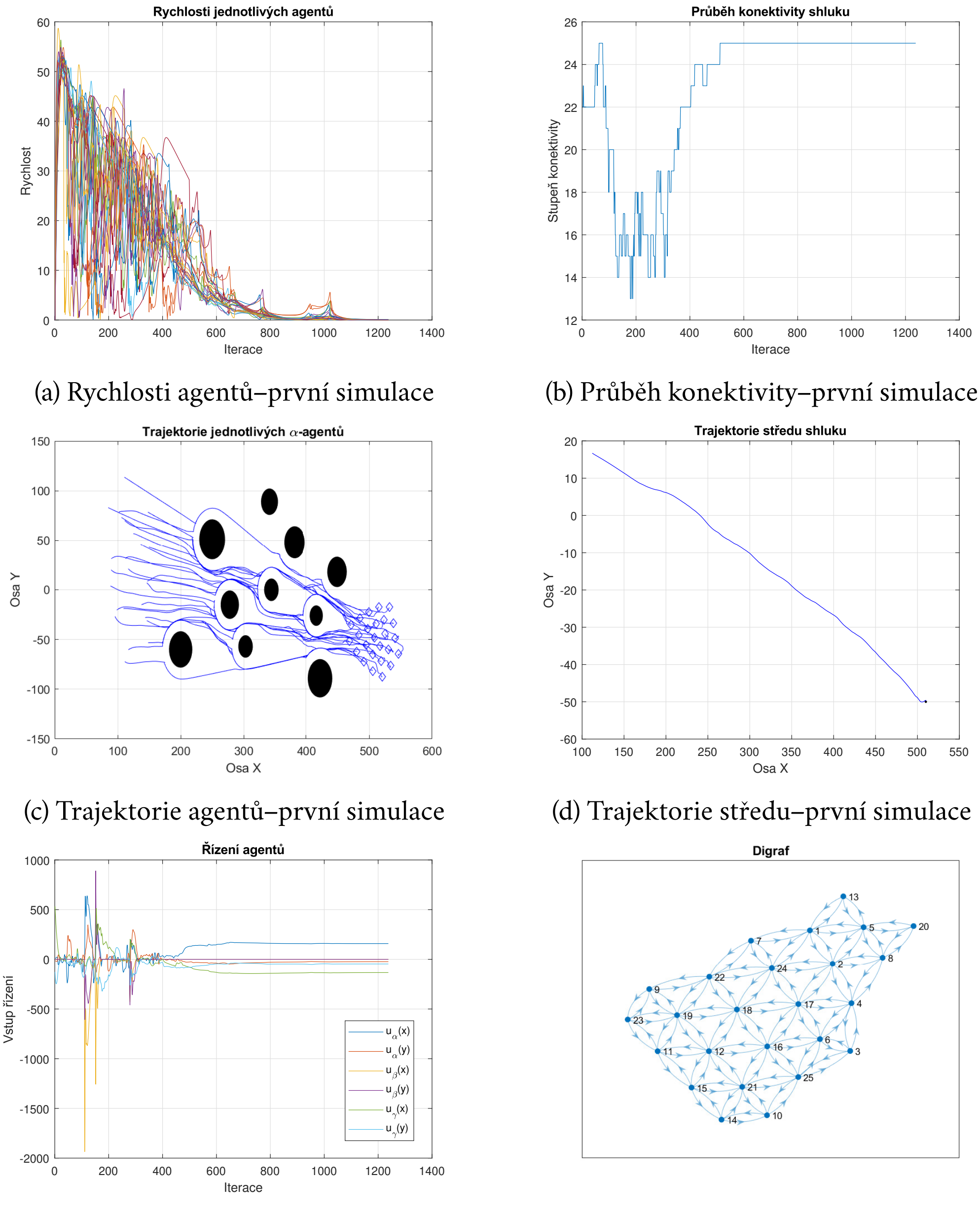
Obrázek 4.5: Počáteční stav–první simulace



Obrázek 4.6: Průběžný stav–první simulace



Obrázek 4.7: Koncový stav–první simulace



(a) Rychlosti agentů–první simulace

(b) Průběh konektivity–první simulace

(c) Trajektorie agentů–první simulace

(d) Trajektorie středu–první simulace

(e) Řízení agentů–první simulace

(f) Konečná formace–první simulace

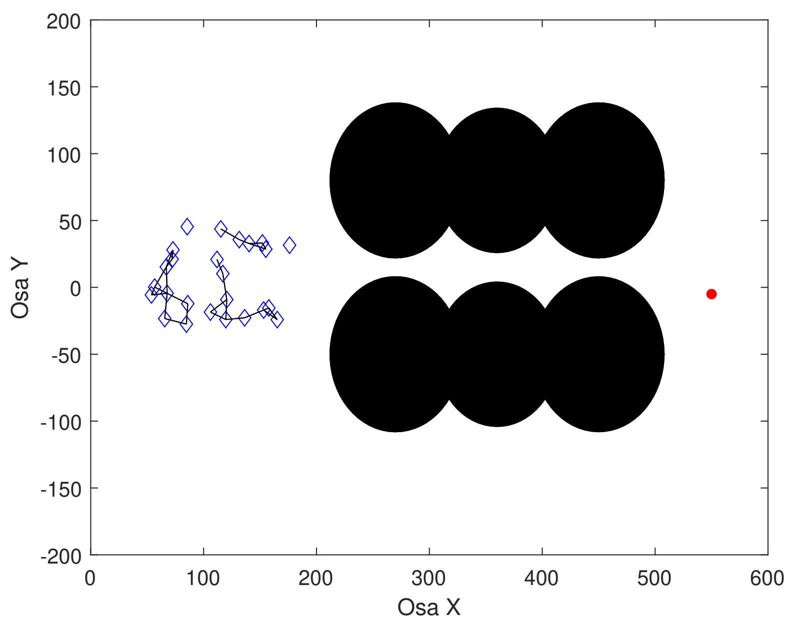
Obrázek 4.8: Výsledky pro první simulaci

4.4.2 2. Simulace–Průchod úzkým prostorem

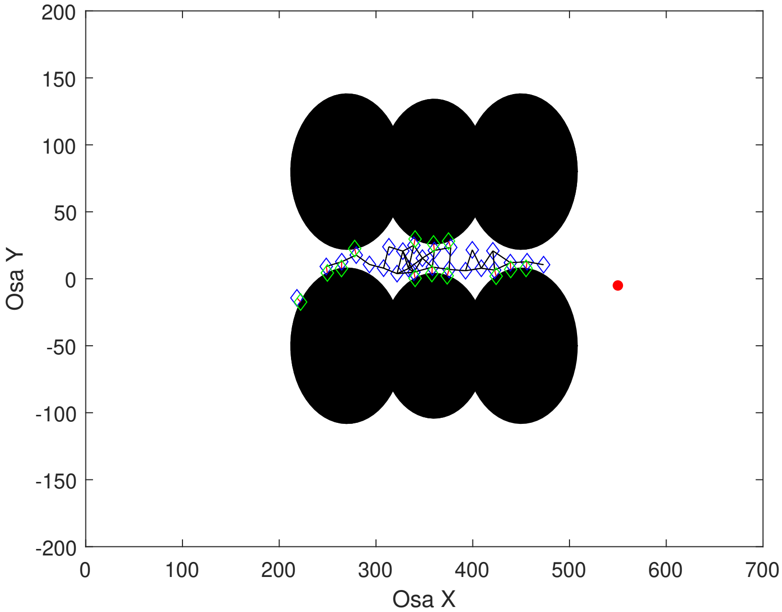
Druhá simulace má za úkol ukázat, jak se algoritmus dokáže vypořádat s řízením a navigací α-agentů úzkým prostorem. Pro počáteční stav této simulace na Obrázku (4.9) byli α-agenti náhodně generováni v rozmezí $(50, 180)$ na ose x a $(-30, 50)$ na ose y, γ-agent má souřadnice $[550, -5]$ a překážky byly vytvořeny následující maticí M:

$$M_2 = \begin{bmatrix} 450 & 450 & 360 & 360 & 270 & 270 \\ -50 & 80 & -50 & 80 & -50 & 80 \\ 58 & 58 & 54 & 54 & 58 & 58 \end{bmatrix}. \tag{4.19}$$

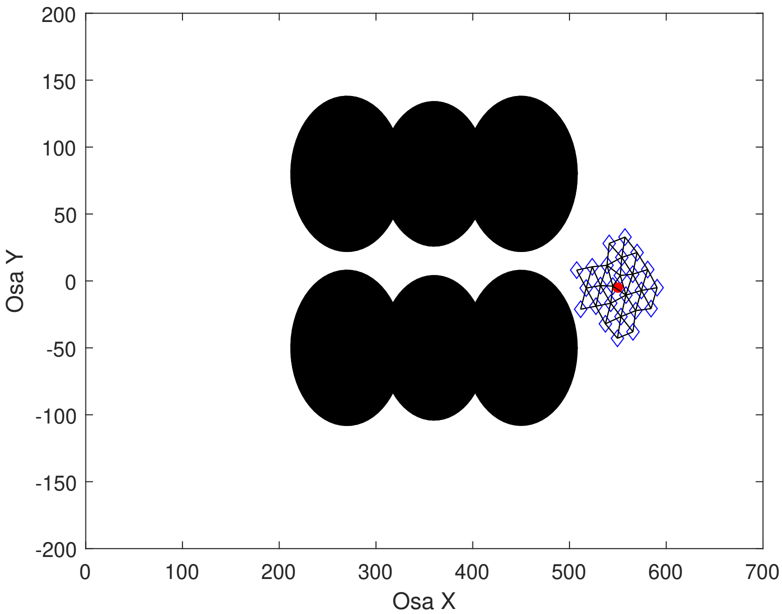
Simulace pak byla spuštěna pro parametry: $n = 25$, $d' = 0.3$, $c_1^\alpha = 35$, $c_1^\beta = 600$, $c_1^\gamma = 22$ a $c_2^\gamma = 2\sqrt{c_1^\gamma}$. Tato simulace má oproti předchozí výrazně menší měřítko vázané mřížky d' proto, aby se snížila interakční vzdálenost mezi α-agenty a β-agenty, což znamená, že se α-agenti drží v menší vzdálenosti od překážek tak, jak je zobrazeno na Obrázku (4.10). Tímto způsobem může algoritmus navigovat α-agenty úzkým prostorem. Je také důležité zmínit, že čím menší je vzdálenost α-agentů od překážky, tím snadněji může dojít ke kolizím. Z tohoto důvodu je nutné snížit i hodnotu parametrů c_1^α a c_1^γ tak, aby se snížila intenzita působení gradientních a konsensuálních termínů. Na Obrázku (4.11) je znázorněn koncový stav simulace a opět tvar výsledného shluku ve tvaru alfa-mřížky. Výsledky na Obrázku (4.12) jsou srovnány v diskusi s ostatními.



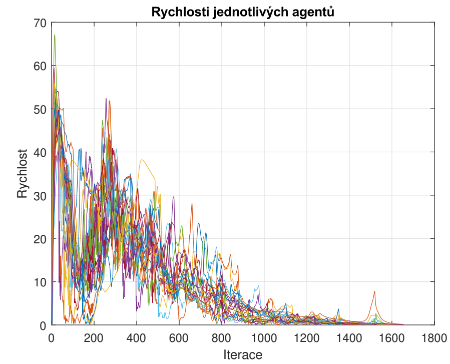
Obrázek 4.9: Počáteční stav–druhá simulace



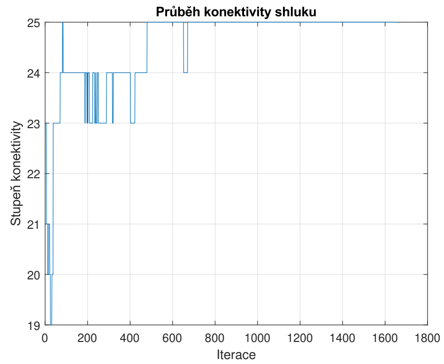
Obrázek 4.10: Průběžný stav–druhá simulace



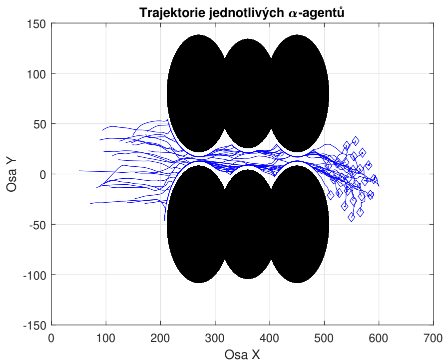
Obrázek 4.11: Koncový stav–druhá simulace



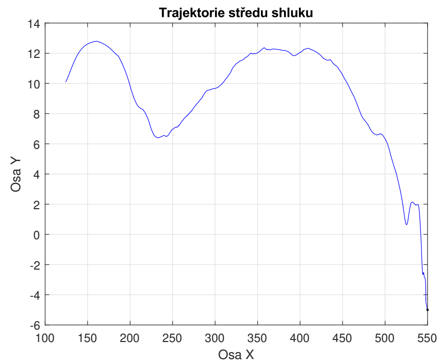
(a) Rychlosti agentů–druhá simulace



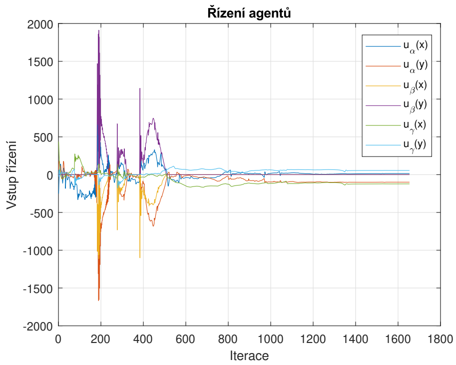
(b) Průběh konektivity–druhá simulace



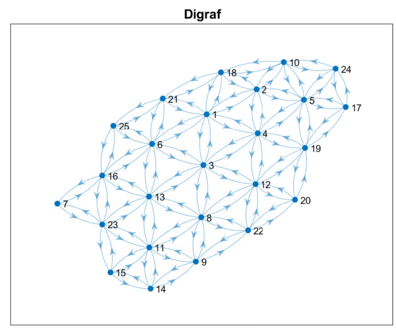
(c) Trajektorie agentů–druhá simulace



(d) Trajektorie středu–druhá simulace



(e) Řízení agentů–druhá simulace



(f) Konečná formace–druhá simulace

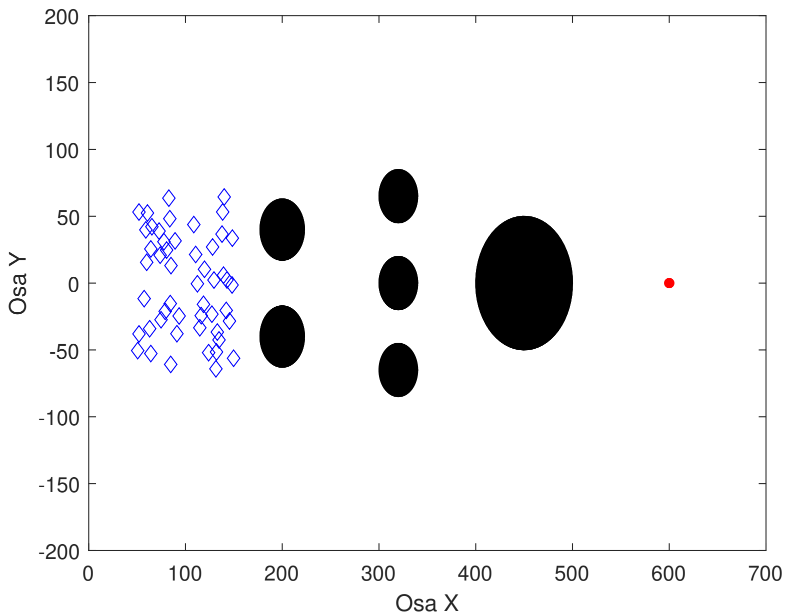
Obrázek 4.12: Výsledky pro druhou simulaci

4.4.3 3. Simulace–Manévr rozdělení a sjednocení

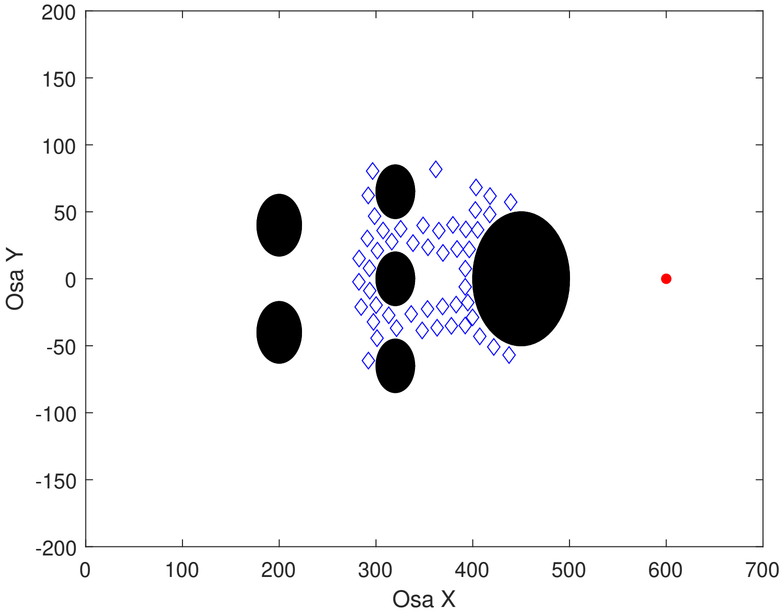
Třetí simulace znázorňuje cílené rozdělení velké skupiny α-agentů a jejich následné sjednocení v cíli skupiny. Pro počáteční stav této simulace na Obrázku (4.13) byli α-agenti náhodně generováni v rozmezí $(50, 150)$ na ose x a $(-65, 65)$ na ose y, γ-agent má souřadnice $[600, 0]$ a překážky byly vytvořeny následující maticí M:

$$M_3 = \begin{bmatrix} 320 & 320 & 320 & 200 & 200 & 450 \\ -65 & 0 & 65 & -40 & 40 & 0 \\ 20 & 20 & 20 & 23 & 23 & 50 \end{bmatrix}. \tag{4.20}$$

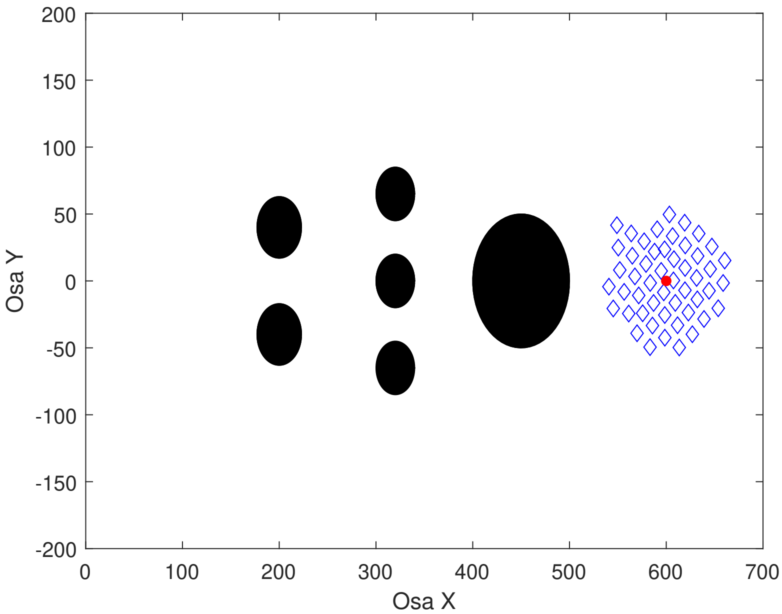
Simulace pak byla spuštěna pro parametry: $n = 50$, $d' = 0.5$, $c_1^\alpha = 45$, $c_1^\beta = 650$, $c_1^\gamma = 21$ a $c_2^\gamma = 2\sqrt{c_1^\gamma}$. Na první pohled se může zdát, že se tato simulace velmi podobá té první. Jde sice o podobný princip, ale překážky zde nejsou vygenerovány náhodně a jsou rozmístěny tak, aby bylo co nejlépe vidět, jak se skupina α-agentů rozděluje o jednotlivé překážky. To je demonstrováno na Obrázku (4.14), kde je možné vidět několik velikostí překážek a chování α-agentů při průchodu mezi nimi. Dalším rozdílem je počet α-agentů, který je oproti té první dvojnásobný. Čím více α-agentů do simulace přidáme, tím vzniká větší výpočetní složitost a simulační program nestíhá vykreslovat všechny hodnoty simulace. Z tohoto důvodu v této simulaci nejsou vykresleny interakční vazby mezi agenty a dále také kvůli přehlednosti, která se také s počtem α-agentů a jejich interakčních vazeb snižuje. Obrázek (4.15) pak opět ukazuje koncový stav simulace a úspěšné vytvoření shluku ve tvar alfa-mřížky. Výsledky simulace na Obrázku (4.16) jsou opět popsány v diskusi výsledků.



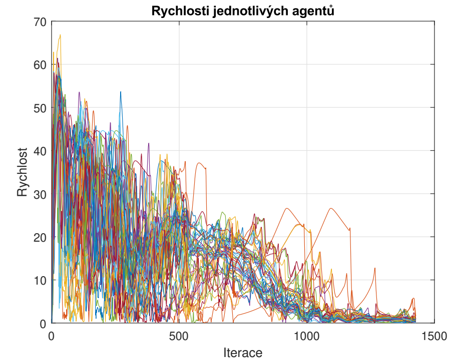
Obrázek 4.13: Počáteční stav–třetí simulace



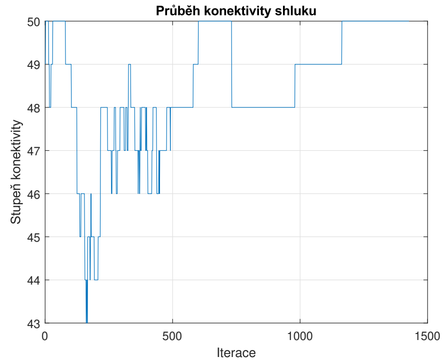
Obrázek 4.14: Průběžný stav–třetí simulace



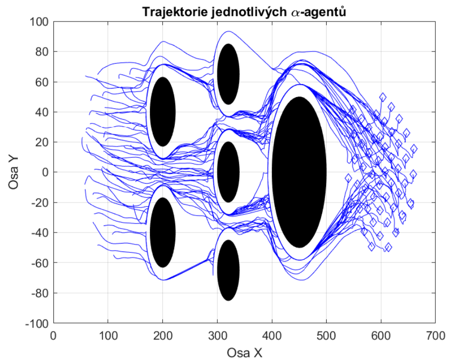
Obrázek 4.15: Koncový stav–třetí simulace



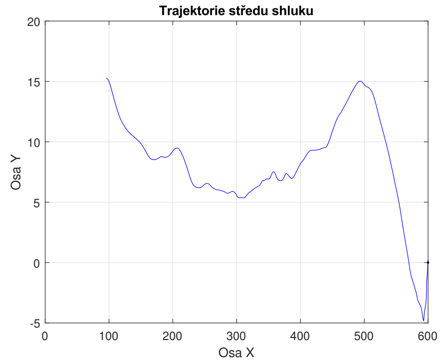
(a) Rychlosti agentů–třetí simulace



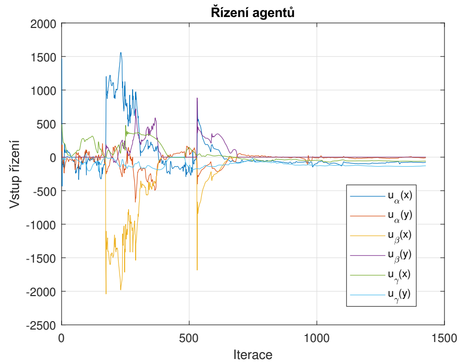
(b) Průběh konektivity–třetí simulace



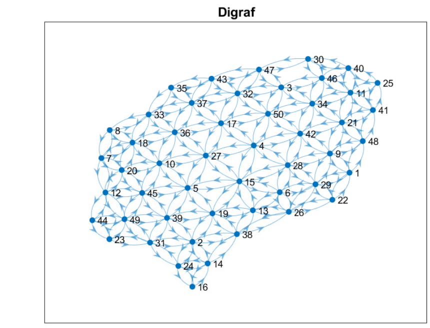
(c) Trajektorie agentů–třetí simulace



(d) Trajektorie středu–třetí simulace



(e) Řízení agentů–třetí simulace



(f) Konečná formace–třetí simulace

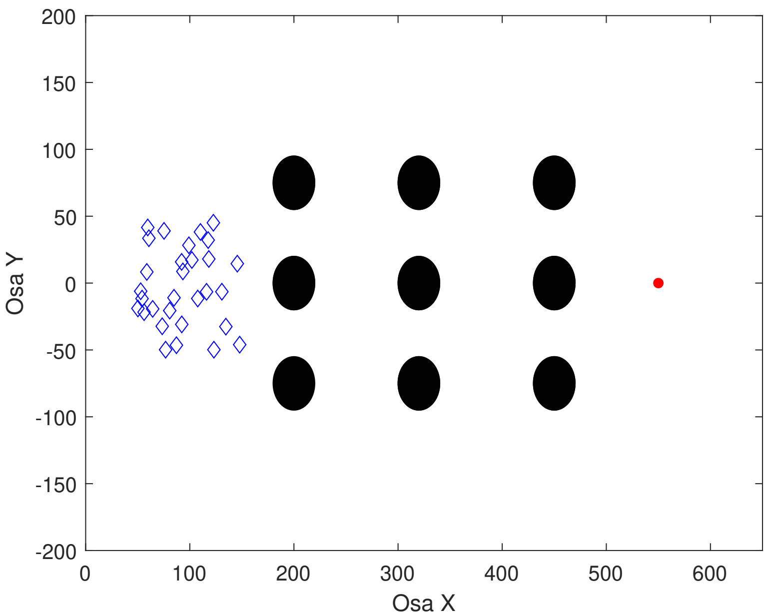
Obrázek 4.16: Výsledky pro třetí simulaci

4.4.4 4. Simulace–Pohybující se γ-agent

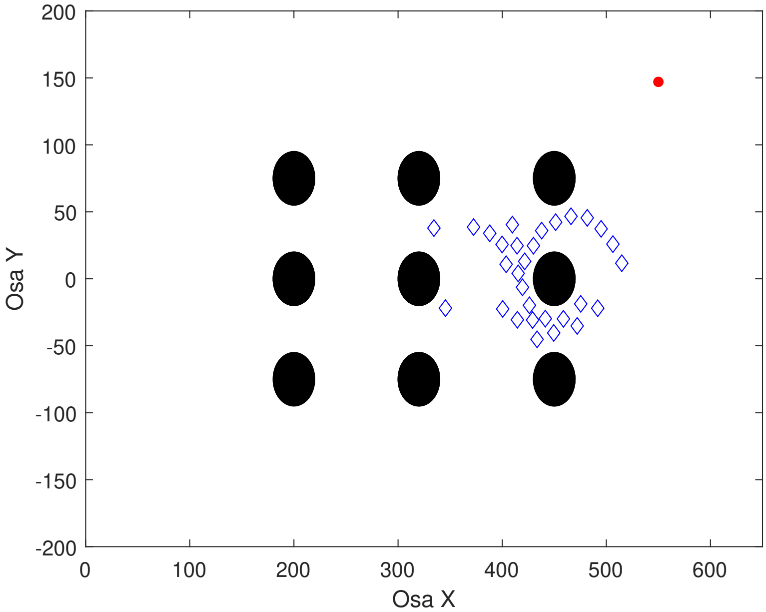
Čtvrtá simulace ukazuje, jak se algoritmus dokáže vypořádat s řízením skupiny α-agentů k cíli skupiny neboli γ-agentovi, který se pohybuje. Pro počáteční stav této simulace na Obrázku (4.17) byli α-agenti náhodně generováni v rozmezí $(50, 150)$ na ose x a $(-50, 50)$ na ose y, γ-agent má souřadnice $[550, 0]$ a překážky byly vytvořeny následující maticí M:

$$M_4 = \begin{bmatrix} 200 & 200 & 200 & 320 & 320 & 320 & 450 & 450 & 450 \\ -75 & 0 & 75 & -75 & 0 & 75 & -75 & 0 & 75 \\ 20 & 20 & 20 & 20 & 20 & 20 & 20 & 20 & 20 \end{bmatrix} \tag{4.21}$$

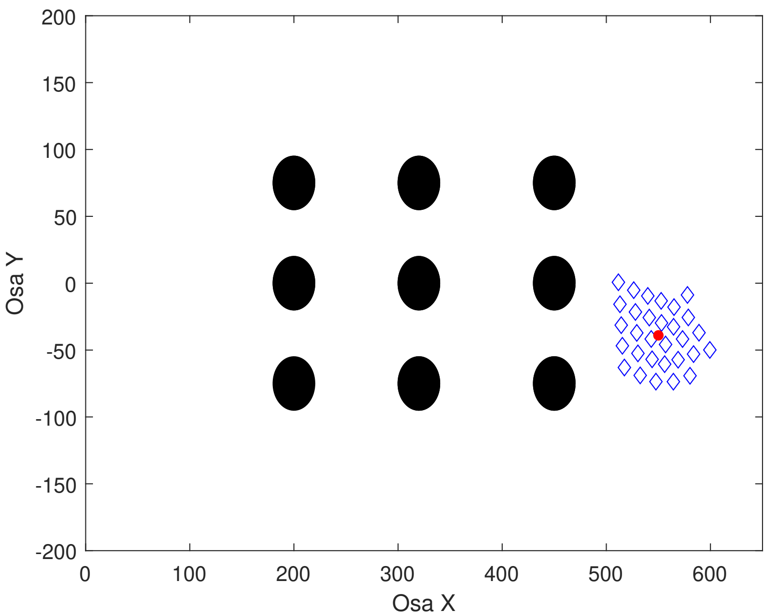
Simulace pak byla spuštěna pro parametry: $n = 30$, $d' = 0.7$, $c_1^\alpha = 30$, $c_1^\beta = 600$, $c_1^\gamma = 20$ a $c_2^\gamma = 2\sqrt{c_1^\gamma}$. γ-agent je nastaven tak, aby se pohyboval nahoru a dolů po ose y na intervalu $(-150, 150)$. Samozřejmě by bylo možné nastavit i složitější pohyb γ-agenta, například pohyb v kruhu. Pro jednoduchou demonstraci sledování cíle však postačí výše uvedené nastavení. Na Obrázku (4.18) je vidět posun γ-agenta z počáteční polohy a snaha α-agentů dosáhnout cíle. Důkaz toho, že α-agenti cíl opravdu sledují je na Obrázku (4.20c), kde je zobrazena trajektorie středu shluku. Jelikož byla pro demonstraci poměrně vysoká rychlost γ-agenta, kvůli které by α-agenti byli schopni cíl pouze sledovat a nikdy nedohnat, byla zde vytvořena zastavovací podmínka. To zastaví pohyb γ-agenta, když α-agenti dosáhnou maximálního stupně konektivity. Na Obrázku (4.19) je pak koncový stav, jež zobrazuje úspěšné dosažení cíle. Výsledky na Obrázku (4.20) jsou opět popsány v diskusi výsledků.



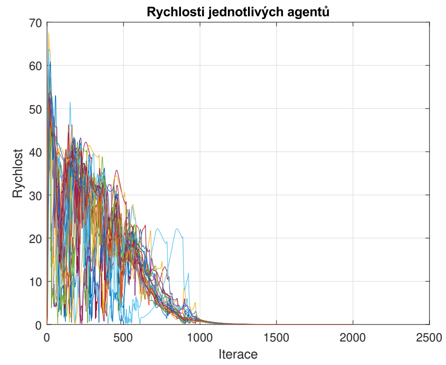
Obrázek 4.17: Počáteční stav–čtvrtá simulace



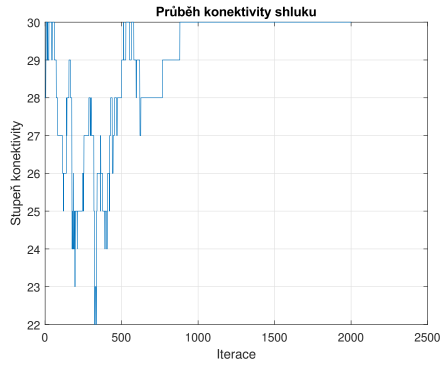
Obrázek 4.18: Průběžný stav–čtvrtá simulace



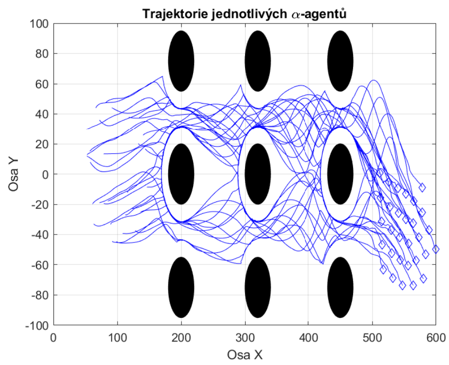
Obrázek 4.19: Koncový stav–čtvrtá simulace



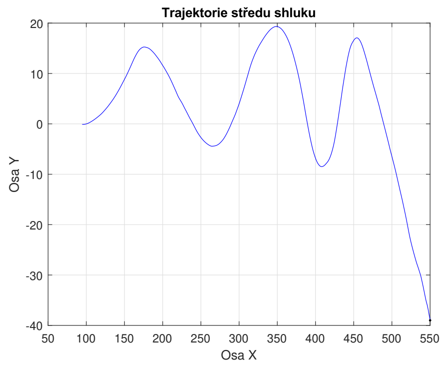
(a) Rychlosti agentů–čtvrtá simulace



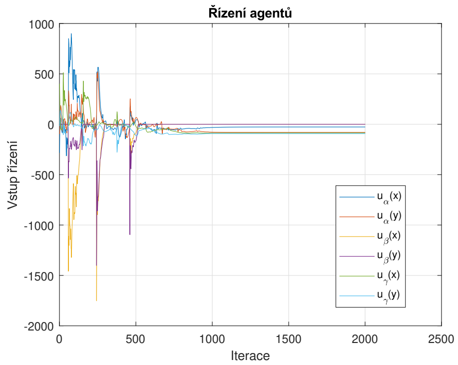
(b) Průběh konektivity–čtvrtá simulace



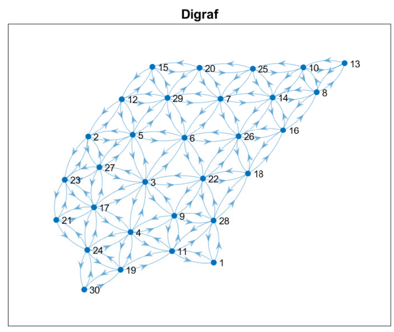
(c) Trajektorie agentů–čtvrtá simulace



(d) Trajektorie středu–čtvrtá simulace



(e) Řízení agentů–čtvrtá simulace



(f) Konečná formace–čtvrtá simulace

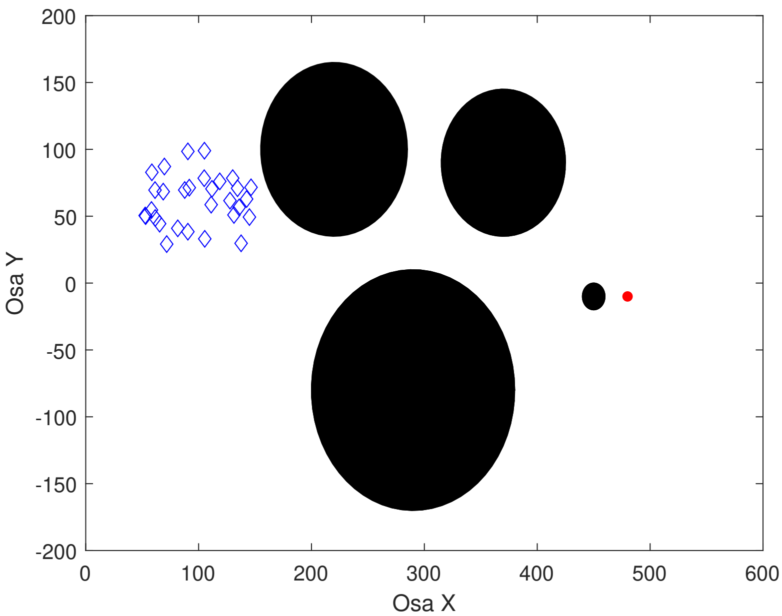
Obrázek 4.20: Výsledky pro čtvrtou simulaci

4.4.5 5. Simulace–Překážka v blízkosti γ-agenta

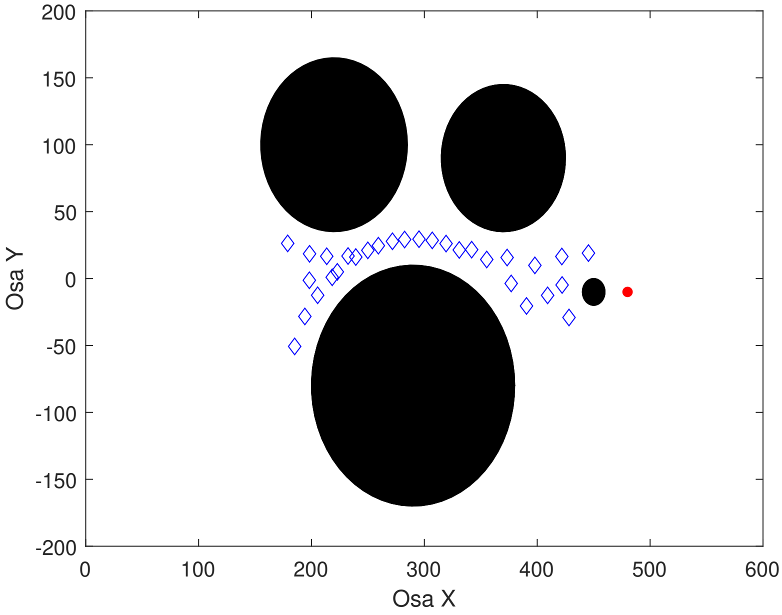
Pátá simulace znázorňuje, jak si algoritmus dokáže poradit s vytvořením shluku α-agentů v cíli skupiny, pokud je tento cíl v blízkosti překážky. Pro počáteční stav této simulace na Obrázku (4.21) byli α-agenti náhodně generováni v rozmezí $(50, 150)$ na ose x a $(20, 100)$ na ose y, γ-agent má souřadnice $[480, -10]$ a překážky byly vytvořeny následující maticí M:

$$M_5 = \begin{bmatrix} 220 & 290 & 370 & 450 \\ 100 & -80 & 90 & -10 \\ 65 & 90 & 55 & 10 \end{bmatrix} \tag{4.22}$$

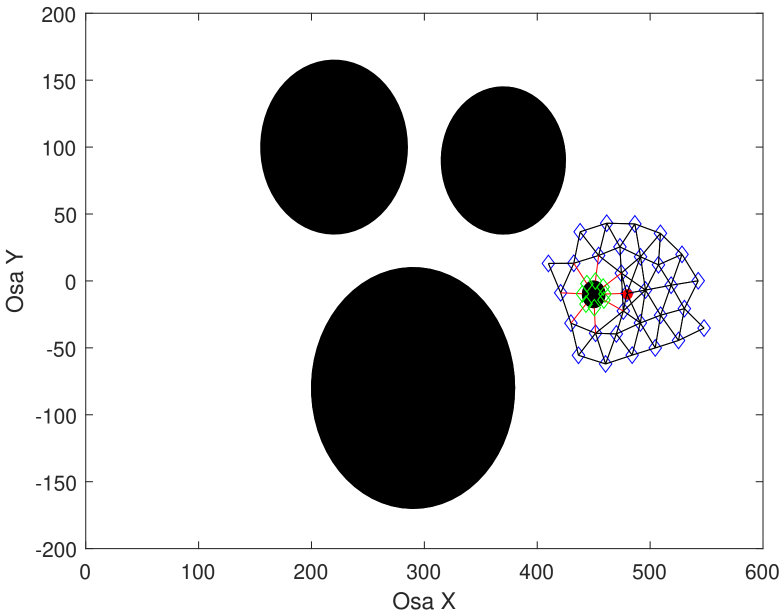
Simulace pak byla spuštěna pro parametry: $n = 30$, zde změníme i parametr $d = 25$, $d' = 0.8$, $c_1^\alpha = 37$, $c_1^\beta = 600$, $c_1^\gamma = 23$ a $c_2^\gamma = 2\sqrt{c_1^\gamma}$. Překážky byly strategicky rozmístěny tak, aby mezi sebou vytvořily úzký prostor. Dále parametry d a d' byly zvětšeny, aby α-agenti byli nejen nuceni vytvořit alfa-mřížku větších rozměrů, ale i držet větší vzdálenost od překážek. Díky tomuto omezení pohybu se α-agenti mezi překážkami pohybují pouze v jedné řadě za sebou, což je vidět na Obrázku (4.22). Díky tomu je stupeň konektivity na Obrázku (4.27b) maximální téměř po celou dobu simulace. Obrázek (4.23) pak znázorňuje konečný stav simulace a výsledný shluk. Jelikož je překážka blízko cíle, α-agenti nevytvoří shluk se středem přesně v cíli, ale vytvoří jej co nejblíže. Výsledný tvar shluku na Obrázku (4.24f) se od ostatních poměrně liší. Důvodem je, že agenti vytvořili shluk kolem překážky, a tak nemá ideální tvar. Výsledky na Obrázku (4.24) jsou opět popsány v diskusi výsledků.



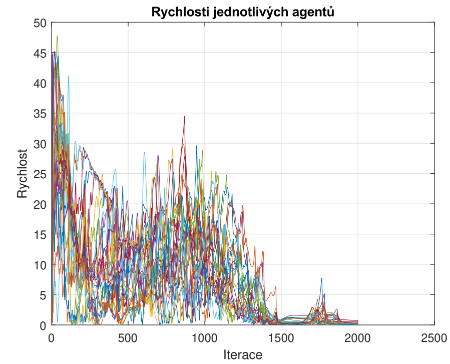
Obrázek 4.21: Počáteční stav–pátá simulace



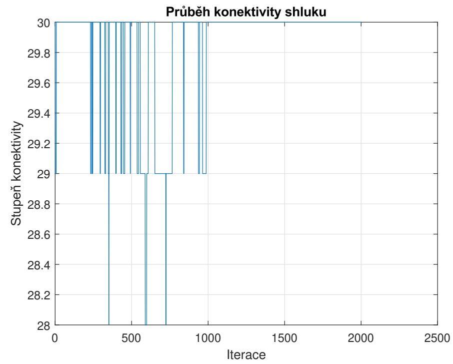
Obrázek 4.22: Průběžný stav–pátá simulace



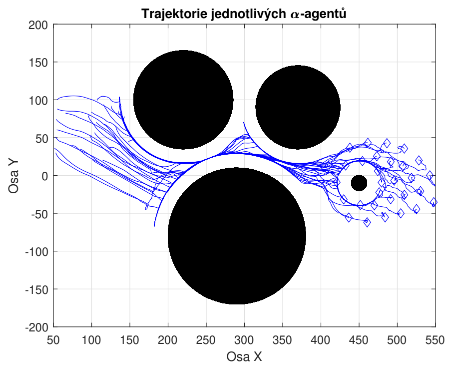
Obrázek 4.23: Koncový stav–pátá simulace



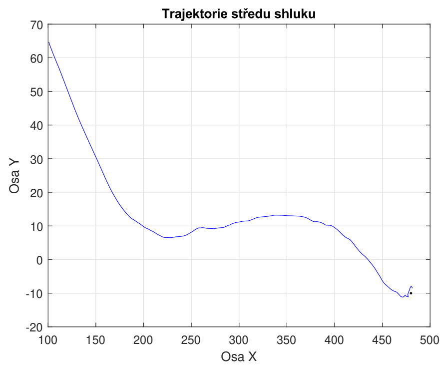
(a) Rychlosti agentů–pátá simulace



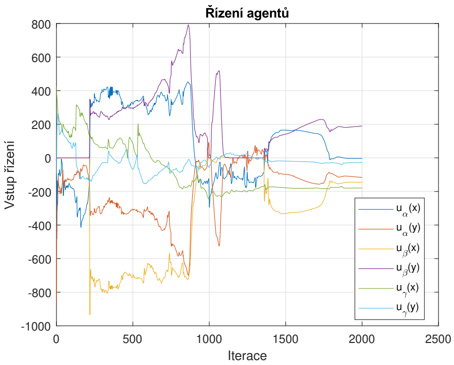
(b) Průběh konektivity–pátá simulace



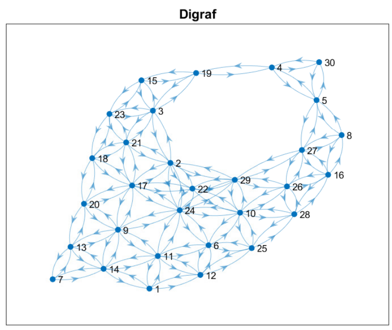
(c) Trajektorie agentů–pátá simulace



(d) Trajektorie středu–pátá simulace



(e) Řízení agentů–pátá simulace



(f) Konečná formace–pátá simulace

Obrázek 4.24: Výsledky pro pátou simulaci

4.4.6 Diskuse výsledků

V této části práce je provedena analýza a porovnání získaných výsledků všech pěti simulací, která ukazuje klíčové rozdíly a podobnosti v chování algoritmu v různých scénářích. Každá simulace trvala jiný počet iterací, jejichž hodnoty jsou následující:

1. Simulace — 1247 iterací
2. Simulace — 1654 iterací
3. Simulace — 1427 iterací
4. Simulace — 1996 iterací
5. Simulace — 2001 iterací

První simulace trvala nejkratší počet iterací a pátá simulace nejdelší, čímž dosáhla i maximálního počtu možných iterací, pro které byla simulace spuštěna. To je způsobeno překážkou v blízkosti γ-agenta, díky které algoritmus nemůže přesně dokonvergovat k cíli. Z tohoto důvodu se simulace teoreticky nikdy nezastaví.

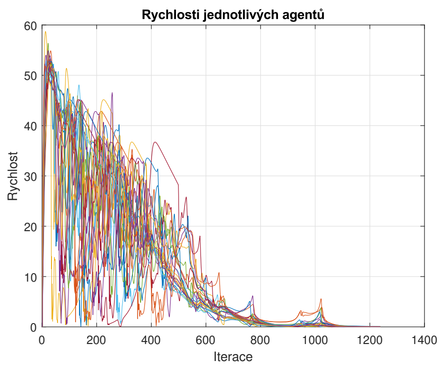
Grafy trajektorií α-agentů a středu shluku není nutné porovnávat, jelikož pouze znázorňují chování α-agentů v průběhu simulace. Každá simulace má jiný průběh, a tak je jejich srovnání irelevantní pro analýzu správné funkčnosti algoritmu. Obdobně je tomu i u digrafů. Pouze v páté simulaci se digraf liší prázdným prostorem bez vazeb mezi α-agenty, což značí přítomnost překážky poblíž cíle skupiny.

Nyní budou porovnány grafy rychlostí na Obrázku (4.25), konektivity na Obrázku (4.26) a řízení jednotlivých simulací na Obrázku (4.27). Rychlosti jednotlivých α-agentů se v průběhu všech simulací snižovaly. U první a druhé simulace je možné vidět poměrně plynulé snižování rychlostí, což naznačuje jednodušší podmínky pro průchod α-agentů kolem překážek. Oproti tomu ve čtvrté a páté simulaci se rychlosti α-agentů opět plynule snižovaly, ale α-agenti měli problém s dosažením cíle skupiny. Ve třetí simulaci pak α-agenti dosáhli cíle skupiny i přes to, že jejich konečné rychlosti byly větší než v ostatních simulacích.

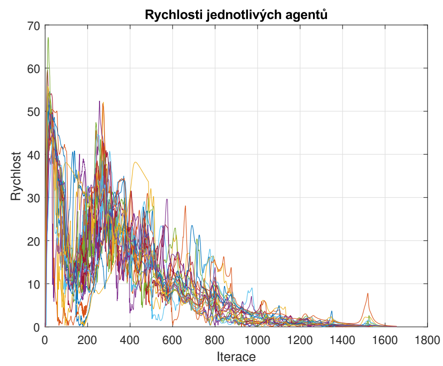
Ve všech simulacích dosáhli α-agenti nejvyššího stupně konektivity, jenž značí správnost tvaru výsledného shluku, což je dále potvrzeno digrafy u každé simulace. Nejlepší průběh konektivity nastal u páté simulace, kde byl její stupeň prakticky většinu času maximální a ve druhé simulaci tomu bylo obdobně. Oproti tomu v první, třetí a čtvrté byl poměrně dlouhou dobu malý. To je způsobeno tím, že tyto simulace obsahovaly překážky, které množinu α-agentů, procházející v okolí překážek, rozdělily do menších skupin. Díky tomu bylo zjištěno, že si algoritmus poradí i se sníženou konektivitou.

Nakonec jsou porovnány řídící vstupy jednotlivých rovnic algoritmu. Všechny řídící složky se ve všech simulacích ustálily kolem nuly, což značí dosažení cíle skupiny a vytvoření výsledného shluku, v němž se α-agenti téměř nepohybují. Dominantními složkami ve všech simulacích jsou $u_\beta(x)$ a $u_\beta(y)$, které ukazují intenzitu odpuzování α-agentů od překážek. Ta je nejvyšší v situacích, kdy je největší počet α-agentů v blízkosti překážek. Z tohoto důvodu jsou tyto složky nenulové na konci páté simulace, což opět značí přítomnost překážky v blízkosti cíle skupiny.

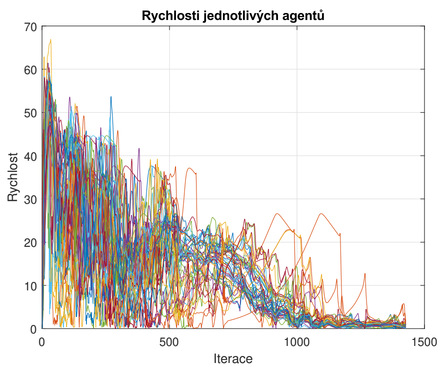
Výsledky získané ze všech typů simulací potvrzují správnou funkčnost a robustnost algoritmu. Dále ukazují, že se algoritmus dokáže vypořádat s různými počty a typy rozložení překážek kruhovitého tvaru, aniž by docházelo ke kolizím.



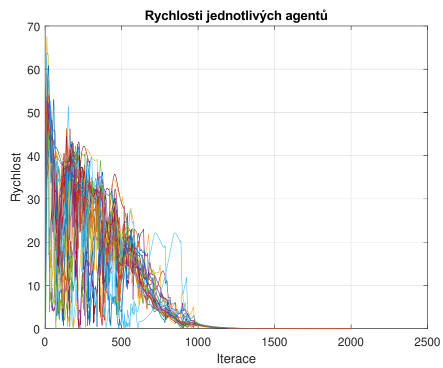
(a) Rychlosti agentů–první simulace



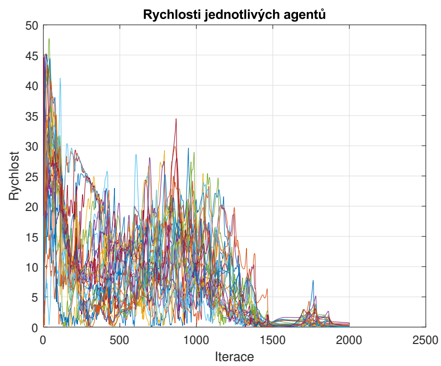
(b) Rychlosti agentů–druhá simulace



(c) Rychlosti agentů–třetí simulace



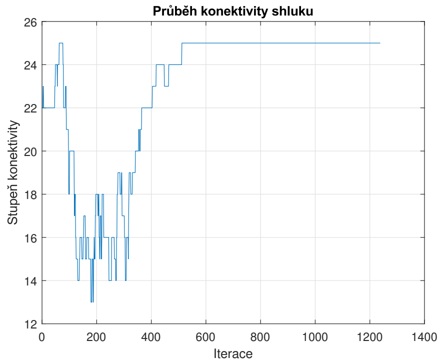
(d) Rychlosti agentů–čtvrtá simulace



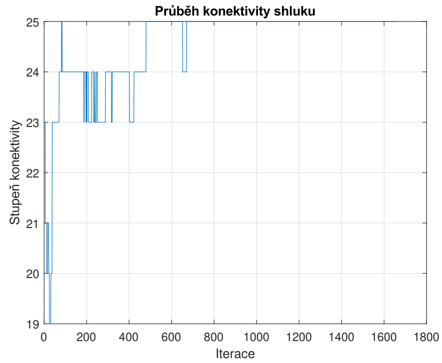
(e) Rychlosti agentů–pátá simulace

Obrázek 4.25: Porovnání rychlostí mezi simulacemi

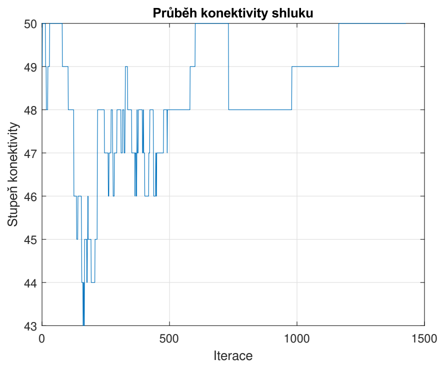
Jednotlivé grafy rychlostí na Obrázku (4.25) nejsou v konkrétních reálných jednotkách, a tak je osa y označena pouze jako *rychlost*. Pro aplikaci algoritmu na reálný svět, je možné rychlosti upravit a vyjádřit téměř kteroukoliv jednotkou rychlosti.



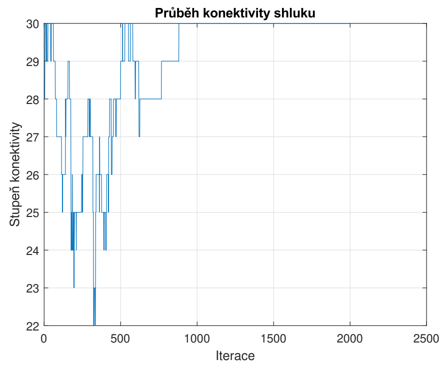
(a) Průběh konektivity–první simulace



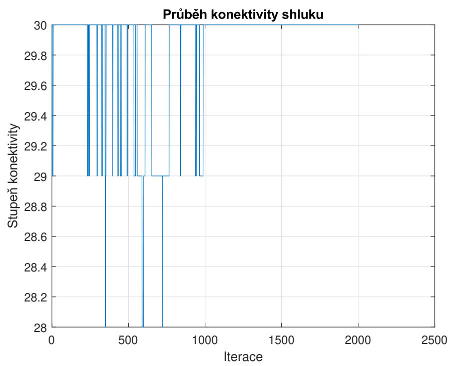
(b) Průběh konektivity–druhá simulace



(c) Průběh konektivity–třetí simulace



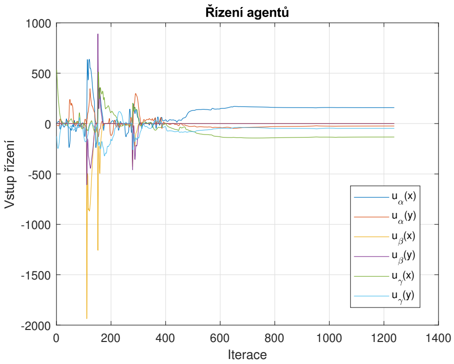
(d) Průběh konektivity–čtvrtá simulace



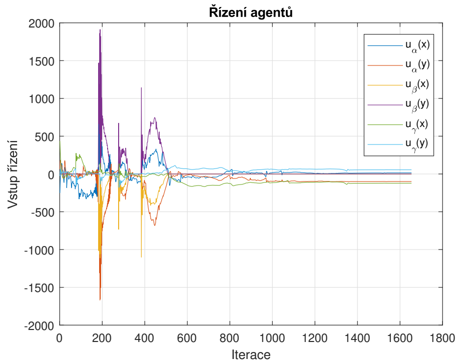
(e) Průběh konektivity–pátá simulace

Obrázek 4.26: Porovnání konektivity mezi simulacemi

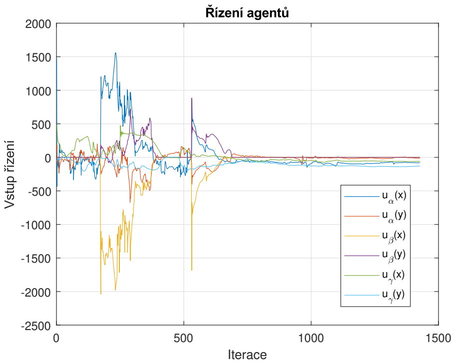
Obdobně je tomu i u grafů řízení na Obrázku (4.27), kde na ose y je *vstup řízení*. Zde také nejsou známé konkrétní jednotky a grafy tedy spíše vizualizují intenzitu působení jednotlivých složek algoritmu.



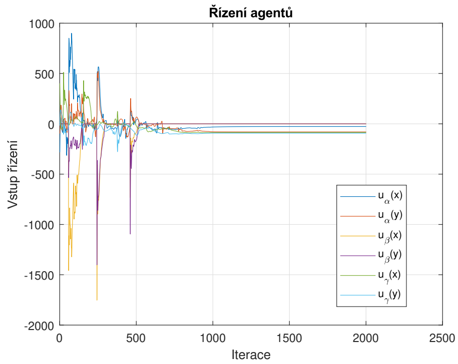
(a) Řízení agentů–první simulace



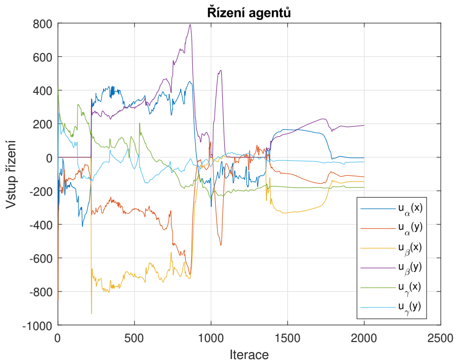
(b) Řízení agentů–druhá simulace



(c) Řízení agentů–třetí simulace



(d) Řízení agentů–čtvrtá simulace



(e) Řízení agentů–pátá simulace

Obrázek 4.27: Porovnání grafů řízení mezi simulacemi

Future works — 5

V rámci dalšího rozvoje této práce lze zkoumat aplikace nových přístupů k reprezentaci jednotlivých agentů. Jedním z možných směrů je vývoj nových algoritmů pro předpovídání a řešení kolizí, které by integrovaly pokročilejší metody strojového učení pro zlepšení rozhodovacích procesů agentů. To by zahrnovalo implementaci hlubokých neuronových sítí pro zlepšení schopnosti agentů předpovídat trajektorie pohybu v reálném čase.

Jako další by bylo možné rozšířit algoritmus o schopnost vytváření a udržování libovolné formace během pohybu α-agentů. Tento přístup by zahrnoval vývoj metod, které by α-agentům umožnily dynamicky měnit formace v reakci na vnější podmínky a překážky. S tím je spojeno využití teorie grafů pro definování nových vztahů mezi jednotlivými typy agentů a aplikování principů decentralizovaného řízení pro rychlou adaptaci formace v reálném čase.

V kontextu praktického nasazení agentů bychom mohli vytvořit simulační modely, které by detailně simulovaly interakce agentů v reálných podmínkách, například v urbanizovaných oblastech nebo v průmyslových komplexech. Tyto modely by mohly být využity pro další analýzu a zlepšení algoritmu, jenž umožňuje detekování a ošetření kolizí.

Jelikož se práce zabývá pouze konvexními překážkami kruhového tvaru, bylo by dobré v budoucnu zkoumat možnosti aplikování algoritmu na jiné typy překážek. Tento výzkum by mohl zahrnovat nekonvexní překážky, překážky s nepravidelnými tvary a dynamické překážky s měnícím se tvarem v čase. Zkoumání těchto možností by umožnilo lepší adaptaci algoritmu na reálné podmínky, kde se mohou agenti setkat s různorodými a komplexními překážkami.

V budoucnu se také nabízí možnost optimalizace komunikačních protokolů mezi agenty, což je klíčové pro zlepšení reakční doby a snížení chyb výpočtů. V simulaci by dále mohlo být zahrnuto několik neurčitostí a následné zkoumání reakcí algoritmu na tyto neurčitosti. Těmi může být například časové zpoždění, zašuměná data, nepřesná detekce vzdálenosti, drop-off paket neboli výpadek informace a další. V reálném světě se totiž není možné těmto neurčitostem vyhnout, čímž se mimo jiné zabývá například tato Práce [10].

Závěr 6

Tato bakalářská práce se zabývala kooperativním řízením multiagentního systému, především pak detekováním a ošetřením kolizí v blízkosti překážek kruhového tvaru. Nejprve se práce věnuje krátkému uvedení do problematiky kooperativního a distribuovaného řízení multiagentních systémů a využití *flockingu* neboli shlukovaní za účelem vytváření formací ve tvaru alfa-mřížky. Byla popsána a vysvětlena grafová teorie včetně uvedení základních pojmů a typů grafů, které mohou být využity v kontextu multiagentních systémů. Grafová teorie pak umožňuje analyzovat vztahy a komunikační vazby mezi agenty, díky nimž je alfa-mřížka definována.

Na základě znalostí z grafové teorie byly definovány jednotlivé typy agentů a vysvětleny funkce, s jejichž pomocí je navrhnut algoritmus pro pohyb a shlukování agentů. Ten se skládá ze tří hlavních částí: gradientní termín, konsensuální termín a navigační zpětná vazba, jejichž funkčnost v algoritmu je také vysvětlena. Nechybí zde ani analýza stability, která ukázala, že systém asymptoticky konverguje k rovnovážnému stavu, při čemž nedochází ke srážkám mezi agenty.

Následující kapitola představuje finální podobu algoritmu, který je rozšířen o schopnost detekce a následné reakce agentů na překážky. Toho je docíleno definováním nových parametrů, které vychází z definic v předešlých kapitolách. Pro ověření funkčnosti algoritmu bylo provedeno pět simulací s různými počátečními podmínkami v podobě rozdílného rozmístění agentů a překážek. Simulace ukázaly, že algoritmus je schopen agenty plynule navigovat kolem několika překážek najednou, úzkým prostorem nebo umožňuje agentům sledovat cíl, při čemž nedochází k žádným srážkám. Nakonec byla diskutována správnost získaných výsledků, jež ukázala, že rychlosti jednotlivých agentů se v průběhu simulace snižují, což potvrzuje konvergenci k rovnovážnému stavu. Dále se zvyšuje stupeň konektivity, čímž je potvrzen výskyt shlukování za účelem vytvoření výsledné formace ve tvaru alfa-mřížky. Výsledky tedy potvrdily správnou funkčnost a robustnost algoritmu, což dále dokazuje i stabilizace řídících složek algoritmu.

Výsledky této bakalářské práce představují přínos k teoretickému i praktickému pochopení kooperativního řízení multiagentních systémů a otevírají nové směry pro možný budoucí výzkum v této rozvíjející se oblasti.

První příloha A

A.1 Inicializace algoritmu

```
%% INICIALIZACE
% Pocet agentu, dimenze, meritko, pomer, reakcni vzdalenost
agents = 30; % <n>
dim = 2; % <dim>
d = 25; % <d>
K = 1.2; % <k>
r = K*d; % <r>

% Rozmisteni agentu a vypocet jejich sousedu
a = 50;
b = 150;
aa = 20;
bb = 100;
x = a + (b-a).*rand(agents,1)
y = aa + (bb-aa).*rand(agents,1)
neighbors = {};
[neighbors] = gen_neighbors(x,y,neighbors,agents,r);

% Prekazka a cil
Ms = [220 290  370 450;    %x
      100 -80  90  -10;    %y
      65  90   55   10];   %radius

target = [480,-10];
connections = 0; % Vizualizace vazeb (1-on,0-off)
tolerance = 0.1; % Parametr zastavovaci podminky
```

```
% Promenne pro vypocet vzdalenosti od prekazky
d_b = 0.8 * d;
r_b = 1.2 * d_b;

% Vykres pocatecni polohy
figure
simulation_plot(x,y,neighbors,target,Ms,agents,connections)
axis([0 600 -200 200])

% Parametry rizeni
epsilon = 0.1; % 0<Epsilon<1
h_alpha = 0.2;
h_beta = 0.7;

c1_alpha = 37;
c2_alpha = 2 * sqrt(c1_alpha);

c1beta = 600;
c2beta = 2 * sqrt(c1beta);

c1_gama = 23;
c2_gama = 2 * sqrt(c1_gama);
% Doba simulace
%---------------------
d_t = 0.03;
t = 0:d_t:60;
%---------------------
```

A.2 Algoritmus

```
%% ALGORITMUS
% Priprava pohybu agentu
old_x = x;
old_y = y;
% Rychlost agentu
nodes_vel1 = zeros(agents,dim);
% Rizeni agentu
u_i = zeros(agents,dim);
```

```
% Trajektorie
x_iter = zeros(agents,length(t));
y_iter = zeros(agents,length(t));
% Rizeni-zasahy
ua_iter = zeros(2,length(t));
ub_iter = zeros(2,length(t));
uc_iter = zeros(2,length(t));
% Trajektorie stredu shluku
center_iteration = zeros(length(t),2);
% Konektivita
con_iter = zeros(length(t),1);
% Jednotlive rychlosti agentu
velocity_it = zeros(length(t),agents);
% Iterace pro graf rychlosti
iter_number = ones(length(t),1);

for iteration = 1:length(t)
    % Vypocet a ukladani rychlosti
    nodes_vel1(:,1) = (x - old_x)/d_t;
    nodes_vel1(:,2) = (y - old_y)/d_t;

    % Kvadrat rychlosti pro graf a pocet iteraci
    if(iteration > 1)
        velocity_it(iteration,:) = sqrt((nodes_vel1
            (:,1).^2)+(nodes_vel1(:,2).^2));
        iter_number(iteration) = iteration;
    end

    % Ukladani predchozi polohy agentu
    old_x = x;
    old_y = y;

    % Vypocet stredu shluku
    xy_center = zeros(1,2);
    xy_center(1) = sum(x)/agents;
    xy_center(2) = sum(y)/agents;

    % Ulozeni pro ukazku prubehu trajektorie
    center_iteration(iteration,:) = xy_center;

```

```
    % Vypocet novych sousedu
    [neighbors] = gen_neighbors(x,y,neighbors,agents,r);

    %Vypocet matice sousednosti + kontrola konektivity
    %<neighbors, agents>
    adj_mat = zeros(agents,agents);
    for i = 1:1:agents
        neigh_len1 = length(neighbors{i});
        for j = 2:1:neigh_len1
            n = neighbors{i}(j);
            adj_mat(i,n) = 1;
        end
    end

    % Ulozeni konektivity jednotlivych iteraci
    con_iter(iteration) = rank(adj_mat);

    % ----------------------------------------------------------

    % Vypocet pozice a rychlosti beta agenta
    % <x, y, obstacle_tran, radius, agents, agent_vel1>
    q_k = zeros(agents,2);
    q_ik = {size(Ms,2)};
    p_ik = {size(Ms,2)};
    I = eye(agents);
    eucld_dist_obst = zeros(agents,1);
    for j = 1:1:size(Ms,2)
        obstacle = Ms(:,j);
        obs_tran = obstacle(1:2);
        obs_radius = obstacle(3);
        for i = 1:1:agents
            eucld_dist_obst(i) = sqrt((obs_tran(1) - x(i))^2 + (obs_tran(2) - y(i))^2);
        end
        mu = obs_radius./eucld_dist_obst;
        a_k = [(x - obs_tran(1))./eucld_dist_obst,(y
```

```
            - obs_tran(2))./eucld_dist_obst];
        P = I - (a_k * a_k');
        p_k = P * nodes_vel1;
        for i = 1:1:agents
            p_k = mu(i) * p_k;
            q_k(i,:) = (mu(i) * [x(i),y(i)]) + ((1 - mu(i)) * obs_tran');
        end
        q_ik{j} = q_k;
        p_ik{j} = p_k;
    end
    % ------------------------------------------------------------
    % Gradientni termin alpha
    % <x, y, distance, it_range, neighbors, agents>
    grad_alpha = zeros(agents,2);
    for i = 1:1:agents
        neigh_len2 = length(neighbors{i});
            nij_val = zeros(agents,2);
            for j = 2:1:neigh_len2
                n1 = neighbors{i}(j);
                eucld_dist_a = sqrt((x(n1) - x(i))^2 + (y(n1) - y(i))^2);
                nij_val(n1,1) = (x(n1) - x(i))/sqrt(1 + (epsilon * (eucld_dist_a^2)));
                nij_val(n1,2) = (y(n1) - y(i))/sqrt(1 + (epsilon * (eucld_dist_a^2)));
            end
        for k = 2:1:neigh_len2
            n2 = neighbors{i}(k);
            z = sqrt((x(n2) - x(i))^2 + (y(n2) - y(i))^2);
            sigma1 = sigma(z);
            grad_alpha(i,:) = grad_alpha(i,:) + (Phi_alpha(sigma1,r,d,h_alpha) * nij_val(n2,:));
        end
    end
    % Konsensualni termin alpha
```

```
    % <nodes_vel1, neighbours, con_mat_a, agents>
    cons_aplha = zeros(agents,2);
    con_mat_a = cons_alpha_mat(x,y,neighbors,r,agents,h_alpha);
    for i = 1:1:agents
        neigh_len3 = length(neighbors{i});
        for j = 2:1:neigh_len3
            n3 = neighbors{i}(j);
            cons_aplha(i,:) = cons_aplha(i,:) + ((con_mat_a(i,n3) * (nodes_vel1(j,:) - nodes_vel1(i,:))));
        end
    end
    % --------------------------------------------------------------

    % Gradientni termin beta
    % <x, y, q_k, r_beta, d_beta, agents>
    grad_beta1 = zeros(agents,2);
    grad_beta = zeros(agents,2);
    nik_val = zeros(agents,2);
    for k = 1:1:length(q_ik)
        q_1 = q_ik{k};
            for i = 1:1:agents
                eucld_dist_b = sqrt((q_1(i,1) - x(i))^2 + (q_1(i,2) - y(i))^2);
                nik_val(i,1) = (q_1(i,1) - x(i))/sqrt(1 + (epsilon * (eucld_dist_b^2)));
                nik_val(i,2) = (q_1(i,2) - y(i))/sqrt(1 + (epsilon * (eucld_dist_b^2)));
            end
        for j = 1:1:agents
            z = sqrt((x(j) - q_1(j,1))^2 + (y(j) - q_1(j,2))^2);
            sig1 = sigma(z);
            grad_beta1(j,:) = Phi_beta(sig1,d_b,h_beta) * nik_val(j,:);
        end
```

```
        grad_beta = grad_beta + grad_beta1;
    end
    % Konsensualni termin beta
    % <node_vel1, p_k, con_mat_b, agents>
    cons_beta = zeros(agents,2);
    con_mat_b = cons_beta_mat(x, y, q_ik, d_b, agents, h_beta);
    for j = 1:1:length(p_ik)
        p_k = p_ik{j};
        for i = 1:1:agents
            cons_beta(i,:) = cons_beta(i,:) + con_mat_b(i,j) * (p_k(i,:) - nodes_vel1(i,:));
        end
    end
    % ----------------------------------------------------------------

    % Upraveni gradientu a konsensu o doporucene parametry
    grad_alpha = c1_alpha * grad_alpha;
    cons_aplha = c2_alpha * cons_aplha;
    grad_beta = c1beta * grad_beta;
    cons_beta = c2beta * cons_beta;

    % Vypocet zmeny pozice vuci cili
    position_dif = zeros(agents,2);
    position_dif(:,1) = (x - target(1));
    position_dif(:,2) = (y - target(2));

    % Vypocet rizeni
    u_aplha = grad_alpha + cons_aplha;
    u_beta = grad_beta + cons_beta;
    sig_gamma = sigma_1(position_dif);
    u_gama1 = (c1_gama * sig_gamma);
    u_gama2 = (c2_gama * nodes_vel1);
    u_i = u_aplha + u_beta - u_gama1 - u_gama2;

    % Zmena pozice agentu
    x = old_x + (d_t * nodes_vel1(:,1)) + (((d_t^2)
```

```
/2) * u_i(:,1));
    y = old_y + (d_t * nodes_vel1(:,2)) + (((d_t^2)/2) * u_i(:,2));

    % Ukladani hodnot kazde iterace
    x_iter(:,iteration) = x;
    y_iter(:,iteration) = y;
    ua_iter(:,iteration) = u_aplha(5,:);
    ub_iter(:,iteration) = u_beta(5,:);
    ug = - u_gama1 - u_gama2;
    uc_iter(:,iteration) = ug(1,:);

    % Aktualizace snimku simulace
    hold off;
    simulation_plot(x,y,neighbors,target,Ms,agents,connections)
    hold on
    if(connections == 1)
    beta_plot(q_ik,r_b,x,y,agents)
    end
    axis([0 600 -200 200])
    hold off;
    drawnow;

    % Zastavovaci podminka
    if(xy_center(1) > target(1)-tolerance && xy_center(1) < target(1)+tolerance && xy_center(2) > target(2)-tolerance && xy_center(2) < target(2)+tolerance && rank(adj_mat) == agents)
        sim_end = iteration;
        break
    else
        sim_end = iteration;
    end

end
```

A.3 Vykreslení výsledků

```
%% Prubeh narazove funkce
figure
values = 0:0.01:2;
ph_values = zeros(1,length(values));
for i = 1:length(values)
    ph_values(i) = ph(values(i),h_alpha);
end
plot(values,ph_values)
hold on
for i = 1:length(values)
    ph_values(i) = ph(values(i),h_beta);
end
plot(values,ph_values)
grid on
box on
title("Narazova funkce")
xlabel("z")
ylabel("\rho_h(z)")
legend("Narazova funkce (h=0.2)","Narazova funkce (h
    =0.7)")

%% Graf akcni funkce
figure
values1 = 1:0.1:40;
phi_alpha_values = zeros(1,length(values1));
for i = 1:length(values1)
    phi_alpha_values(i) = Phi_alpha(values1(i)
        ,1.2*10,10,h_alpha);
end
plot(values1,phi_alpha_values)
hold on
for i = 1:length(values1)
    phi_alpha_values(i) = Phi_alpha(values1(i)
        ,1.2*10,10,h_beta);
end
plot(values1,phi_alpha_values)
grid on
```

```
box on
title("Akcni funkce")
xlabel("z")
ylabel("\phi_\alpha(z)")
legend("Akcni funkce (h=0.2)","Akcni funkce (h=0.7)")
%% Graf pro rychlosti agentu
figure
for i = 1:1:agents
    plot(iter_number(1:sim_end,1), velocity_it(1:sim_end,i));
    title('Rychlosti jednotlivych agentu');
    hold on;
    grid on;
    box on;
    set(gcf,'color','w');
end
xlabel("Iterace")
ylabel("Rychlost")
%% Graf stupne konektivity
figure
plot(iter_number(1:sim_end,1), con_iter(1:sim_end,1));
grid on
box on
set(gcf,'color','w');
title('Prubeh konektivity shluku');
xlabel("Iterace")
ylabel("Stupen konektivity")
%% Graf trajektorii agentu
figure
scatter(x_iter(sim_end,:),y_iter(sim_end,:),'bluediamond');
hold on
for i = 1:1:agents
    plot(x_iter(1:sim_end,i),y_iter(1:sim_end,i),'b');
    hold on
     phi = 0:.1:2*pi;
    for o =1:size(Ms,2)
        k = Ms(3,o) * cos(phi);
```

```
            l = Ms(3,0) * sin(phi);
            plot(k + Ms(1,0),l + Ms(2,0),'black')
            fill(k + Ms(1,0),l + Ms(2,0),'black')
            hold on;
        end
        box on
        grid on
        set(gcf,'color','w');
end
title('Trajektorie jednotlivych \alpha-agentu')
xlabel("Osa X")
ylabel("Osa Y")
%% Graf trajektorie stredu shluku
figure
plot(center_iteration(1:sim_end,1),center_iteration(1:sim_end,2),'b-');
hold on
plot(target(1),target(2),'k.');
title('Trajektorie stredu shluku')
xlabel("Osa X")
ylabel("Osa Y")
hold on
grid on
box on
set(gcf,'color','w');
%% Graf rizeni agentu
figure
hold on
plot(iter_number(1:sim_end),ua_iter(1,1:sim_end))
plot(iter_number(1:sim_end),ua_iter(2,1:sim_end))

plot(iter_number(1:sim_end),ub_iter(1,1:sim_end))
plot(iter_number(1:sim_end),ub_iter(2,1:sim_end))

plot(iter_number(1:sim_end),uc_iter(1,1:sim_end))
plot(iter_number(1:sim_end),uc_iter(2,1:sim_end))
title("Rizeni agentu")
xlabel("Iterace")
ylabel("Vstup rizeni")
legend("u_\alpha(x)","u_\alpha(y)", "u_\beta(x)","u_\
```

```
    beta(y)", "u_\gamma(x)", "u_\gamma(y)")
grid on
box on
set(gcf,'color','w');
%% Tvar vysledneho shluku
figure
plot(digraph(adj_mat))
title("Digraf")
set(gcf,'color','w');
```

A.4 Odkaz na celé řešení

Na Obrázku (A.1) je odkaz formou QR kódu, který vede na GitHub, kde jsou nahrané a volně dostupné všechny zdrojové soubory.

Obrázek A.1: Odkaz na GitHub

Bibliografie

[1] Aizaz U. Chaudhry a Halim Yanikomeroglu. „Laser Intersatellite Links in a Starlink Constellation: A Classification and Analysis". In: *IEEE Vehicular Technology Magazine* 16.2 (2021), s. 48–56. DOI: 10.1109/MVT.2021.3063706.

[2] Jiajia Chen, Zheng Zhou, Yue Duan a Biao Yu. „Research on Reinforcement-Learning-Based Truck Platooning Control Strategies in Highway On-Ramp Regions". In: *World Electric Vehicle Journal* 14.10 (2023). ISSN: 2032-6653. URL: https://www.mdpi.com/2032-6653/14/10/273.

[3] J. de Curtò, I. de Zarzà, Juan Carlos Cano, Pietro Manzoni a Carlos T. Calafate. „Adaptive Truck Platooning with Drones: A Decentralized Approach for Highway Monitoring". In: *Electronics* 12.24 (2023). ISSN: 2079-9292. URL: https://www.mdpi.com/2079-9292/12/24/4913.

[4] Adelinde M Uhrmacher a Danny Weyns. *Multi-Agent systems: Simulation and applications*. CRC press, 2009.

[5] John Von Neumann a Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton, NJ, USA: Princeton University Press, 1944.

[6] Arnoštka Netrvalová. „Úvod do problematiky multiagentních systémů". In: *West Bohemia University* (). URL: https://www.kiv.zcu.cz/~netrvalo/phd/MAS.pdf.

[7] J.A. Fax a R.M. Murray. „Information flow and cooperative control of vehicle formations". In: *IEEE Transactions on Automatic Control* 49.9 (2004), s. 1465–1476. DOI: 10.1109/TAC.2004.834433.

[8] Joseph Alexander Fax. *Optimal and cooperative control of vehicle formations*. California Institute of Technology, 2002.

[9] Ping Xuan a Victor Lesser. „Multi-agent policies: from centralized ones to decentralized ones". In: AAMAS '02. Bologna, Italy: Association for Computing Machinery, 2002, s. 1098–1105. ISBN: 1581134800. DOI: 10.1145/545056.545078. URL: https://doi.org/10.1145/545056.545078.

[10] Karel Kubíček a Jindřich Wolf. „Distributed method for Economic Dispatch Problem in power network with multiple uncertainties“. In: *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*. 2022, s. 1–8. DOI: 10.1109/ETFA52439.2022.9921437.

[11] Karel Kubicek, Martin Cech a Martin Strelec. „A Robust Distributed Algorithm for Solving the Economic Dispatch Problem with the Penetration of Renewables and Battery Systems“. In: *Applied Sciences* 14.5 (2024). ISSN: 2076-3417. DOI: 10.3390/app14051991. URL: https://www.mdpi.com/2076-3417/14/5/1991.

[12] Mariacristina Roscia, Michela Longo a George Cristian Lazaroiu. „Smart City by multi-agent systems“. In: *2013 International Conference on Renewable Energy Research and Applications (ICRERA)*. 2013, s. 371–376. DOI: 10.1109/ICRERA.2013.6749783.

[13] Frans CA Groen, Matthijs TJ Spaan, Jelle R Kok a Gregor Pavlin. „Real world multi-agent systems: information sharing, coordination and planning“. In: *Logic, Language, and Computation: 6th International Tbilisi Symposium on Logic, Language, and Computation, TbiLLC 2005 Batumi, Georgia, September 12-16, 2005. Revised Selected Papers 6*. Springer. 2007, s. 154–165.

[14] Jindřich Wolf. „Řízení kolaborativních multi-agentních dynamických systémů“. diplomathesis. 2019.

[15] Jesús Ibáñez, Antonio F. Gómez-Skarmeta a Josep Blat. „DJ-boids: emergent collective behavior as multichannel radio station programming“. In: *Proceedings of the 8th International Conference on Intelligent User Interfaces*. IUI ’03. Miami, Florida, USA: Association for Computing Machinery, 2003, s. 248–250. ISBN: 1581135866. DOI: 10.1145/604045.604089. URL: https://doi.org/10.1145/604045.604089.

[16] Craig W. Reynolds. „Flocks, herds, and schools: a distributed behavioral model“. In: *Seminal Graphics: Pioneering Efforts That Shaped the Field, Volume 1*. New York, NY, USA: Association for Computing Machinery, 1998, s. 273–282. ISBN: 158113052X. URL: https://doi.org/10.1145/280811.281008.

[17] Robin J Wilson. *Introduction to graph theory*. USA: John Wiley & Sons, Inc., 1986. ISBN: 0470206160.

[18] Mehran Mesbahi a Magnus Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. STU - Student edition. Princeton University Press, 2010. ISBN: 9780691140612. URL: http://www.jstor.org/stable/j.ctt1287k9b (cit. 09. 04. 2024).

[19] R. Olfati-Saber. „Flocking for multi-agent dynamic systems: algorithms and theory“. In: *IEEE Transactions on Automatic Control* 51.3 (2006), s. 401–420. DOI: `10.1109/TAC.2005.864190`.

[20] Reza Olfati-Saber a RM Murray. „Flocking with obstacle avoidance“. In: *California Inst. Technol., Control Dyna. Syst., Pasadena, CA, Tech. Rep. CIT-CDS* (2003).

[21] R. Olfati-Saber a R.M. Murray. „Consensus problems in networks of agents with switching topology and time-delays“. In: *IEEE Transactions on Automatic Control* 49.9 (2004), s. 1520–1533. DOI: `10.1109/TAC.2004.834113`.

[22] Karel Kubicek a Lukáš Bláha. „Nelineární systémy, podklady k přednáškám“. In: *West Bohemia University* (2023).

[23] Brian L. Partridge. „The Structure and Function of Fish Schools“. In: *Scientific American* 246.6 (1982), s. 114–123. ISSN: 00368733, 19467087. URL: `http://www.jstor.org/stable/24966618` (cit. 22. 04. 2024).

Seznam obrázků