|  |
| --- |
| **Zápočtový program – dokumentace** |
|  |
| Swarm Intelligence (Boids) |
|  |
| Lukáš Stibor |
|  |
| **2025** |

Obsah

[1 Uživatelská dokumentace 1](#_Toc207920064)

[1.1 Účel aplikace 1](#_Toc207920065)

[1.2 Instalace 1](#_Toc207920066)

[1.2.1 Požadavky na systém 1](#_Toc207920067)

[1.2.2 Postup instalace 1](#_Toc207920068)

[1.2.3 Soubory aplikace 1](#_Toc207920069)

[1.3 Ovládání a interakce 1](#_Toc207920070)

[1.4 Ukončení aplikace 2](#_Toc207920071)

[2 Programátorská dokumentace 3](#_Toc207920072)

[2.1 Top Level 3](#_Toc207920073)

[2.1.1 Specifikace problému 3](#_Toc207920074)

[2.1.2 Algoritmické řešení 3](#_Toc207920075)

[2.1.3 Architektura a dekompozice 4](#_Toc207920076)

[2.2 Low Level 4](#_Toc207920077)

[2.2.1 MainWindow 4](#_Toc207920078)

[2.2.2 Boid 5](#_Toc207920079)

[2.2.3 SimulationState 5](#_Toc207920080)

[2.2.4 BoidData 5](#_Toc207920081)

[2.2.5 ImportExport 5](#_Toc207920082)

[2.3 Testování 5](#_Toc207920083)

[2.3.1 Test 1: Vyrovnané hejno v trojúhelníku (základní chování) 6](#_Toc207920084)

[2.3.2 Test 2: Blízko umístěné boidy (test vyhýbání kolizím) 6](#_Toc207920085)

[2.3.3 Test 3: Boidy s různými směry (test zarovnání) 7](#_Toc207920086)

[2.3.4 Test 4: Rozptýlené hejno (test soudržnosti) 8](#_Toc207920087)

[2.3.5 Test 5: Náhodné hejno 8](#_Toc207920088)

[Závěr 10](#_Toc207920089)

[Zdroje 11](#_Toc207920090)

# Uživatelská dokumentace

## Účel aplikace

Cílem programu je vytvořit simulaci kolektivního chování hejna pomocí modelu "boids", který je inspirován přirozeným pohybem ptáků nebo ryb v hejnu. Jednoduší agenti (tzv. boidi) se pohybují podle několika jednoduchých pravidel a společně vykazují komplexní skupinové chování – tzv. swarm intelligence.

Simulace probíhá v reálném čase a uživatel má možnost měnit parametry tak, aby mohl okamžitě pozorovat, jak jednotlivá nastavení ovlivňují chování celého hejna.

## Instalace

### Požadavky na systém

Pro spuštění aplikace je potřeba operační systém **Windows 10** nebo **Windows 11** a nainstalované prostředí **.NET 7.0 Runtime** nebo novější. Pokud není na počítači dostupné, lze ho zdarma stáhnout z oficiálních stránek *dotnet.microsoft.com*.

### Postup instalace

Aplikaci nainstalujete, tak že stáhnete připravenou složku s programem. Po stažení složku rozbalte do libovolného umístění na disku.

### Soubory aplikace

Hlavní soubor je spustitelný soubor **boids.exe**. Spolu s ním se ve složce nacházejí další soubory, které pro uživatele nejsou důležité, avšak jsou důležité pro zajištění ladného běhu programu.

## Ovládání a interakce

Běh simulace lze kdykoliv spustit nebo zastavit v sekci CONTROLS pomocí tlačítka **Start/Stop**. Tlačítkem **Reset** můžete resetovat pozice boidů, na nové, zcela náhodné, pozice. Tlačítko **Import** slouží k načtení souboru s daty o boidech a jejich konfiguraci. **Export** naopak uloží do zařízení soubor s těmito daty o právě nastavené simulaci a jejich boidech.

Pod tlačítky se nachází sekce PARAMETERS obsahující posuvníky (slidery), kterými lze měnit klíčové parametry simulace:

* BOID COUNT – počet boidů v simulaci (aktualizuje se po resetu)
* SEPARATION – síla vyhýbání se ostatním boidům
* ALIGMENT – síla, jak moc se boid orientuje podle směru ostatních v jeho okolí
* COHESION – míra soudržnosti hejna (boid se snaží držet blízko středu svého okolního hejna)

Poslední sekce OPTIONS umožňuje nastavit zobrazování tzv. trailu. Trail je pouze vizuální efekt pro lepší pochopení proudění hejna. Často ale může omezovat, jelikož je výpočetně náročnější. Slider MAX SPEED nastavuje maximální rychlost, které boid může dosáhnout.

V pravém dolním rohu simulačního okna je umístěno číslo, vyjadřující počet **FPS** (Frames Per Second), který zobrazuje aktuální počet snímků za sekundu.

## Ukončení aplikace

Aplikaci lze zavřít kliknutím na **křížek** v pravém horním rohu okna. Alternativně můžete použít klávesovou zkratku **Alt + F4**.

# Programátorská dokumentace

## Top Level

### Specifikace problému

Úlohou je simulovat kolektivní chování skupiny autonomních agentů (tzv. boidů) podle modelu Craiga Reynoldse (1986). Tento model je považován za klasický příklad emergentního chování – tedy situace, kdy jednoduchá lokální pravidla vedou ke komplexnímu globálnímu výsledku. Každý boid je popsán polohou a rychlostí a jeho chování je určováno třemi základními pravidly:

* Separation (oddělování) – boid se snaží vyhnout přílišnému přiblížení k jiným boidům, aby nedocházelo ke kolizím.
* Alignment (zarovnání) – boid upravuje svůj směr, aby se přiblížil průměrnému směru sousedních boidů.
* Cohesion (soudržnost) – boid se snaží pohybovat směrem k průměrné poloze svých sousedů, čímž se hejno udržuje pohromadě.

Pro správné fungování modelu je zásadní definovat lokální okolí boida – boid reaguje pouze na ty jedince, kteří leží v určité vzdálenosti a úhlu od jeho aktuálního směru. Boidi mimo tuto oblast jsou ignorováni.

### Algoritmické řešení

Z algoritmického hlediska má základní implementace složitost O(n²), protože každý boid musí vyhodnotit vzdálenost vůči všem ostatním. Efektivnější implementace využívají prostorové indexovací struktury, které zrychlují vyhledávání sousedů na téměř O(n).

1. Pro každý boid:

* Se vypočítají vektory Separation, Alignment a Cohesion na základě sousedních boidů.
* Vektory se zkombinují s váhami (SeparationStrength, AlignmentStrength, CohesionStrength).
* Aktualizuje se rychlost (Velocity) přidáním výsledného zrychlení a omezením na MaxSpeed.
* Aktualizuje se pozice (Position) přidáním rychlosti.

1. Vykreslí se boidy jako otočené trojúhelníky na Canvas podle jejich pozice a směru rychlosti.
2. FPS se aktualizuje každou sekundu na základě počtu snímků (frameCount).

### Architektura a dekompozice

Aplikace je rozdělena do pěti tříd, jak je znázorněno v UML diagramu níže. Architektura odděluje UI, logiku simulace a I/O operace.

* **MainWindow** – třída je zodpovědná za uživatelské rozhraní a celkový běh simulace. Obsahuje seznam boidů, spravuje časomíru a obsluhuje interakce s uživatelem (tlačítka, slidery, checkboxy). Také zajišťuje pravidelné volání aktualizací a vykreslování boidů, počítá a zobrazuje FPS.
* **Boid** – třída reprezentující jednotlivého agenta, tzv. boida. Uchovává jeho pozici, rychlost a parametry chování. Implementuje pravidla Separation, Alignment, Cohesion. Obsahuje metody pro výpočet vektorů a aktualizaci pohybu na základě výsledné akcelerace.
* **SimulationState** – datová třída, která slouží k uchování kompletního stavu simulace. Obsahuje seznam boidů a hodnoty parametrů simulace.
* **BoidData** – jednodušší datová struktura reprezentující pozici a rychlost jednotlivého boida, určená pro ukládání a načítání. Slouží jako most mezi třídou Boid a třídou SimulationState.
* **ImportExport** – pomocná třída zajišťující export a import stavu simulace. Používá serializaci/deserializaci JSON pomocí knihovny System.Text.Json a zpřístupňuje jednoduché metody Export() a Import(), které se jednoduše využívají v hlavním okně aplikace.

## Low Level

### MainWindow

* + - Role: Spravuje UI (Canvas, slidery, tlačítka, checkboxy) a logiku simulace (seznam boidů, timer, FPS).
    - Atributy:
      * boids: List<Boid> – seznam boidů.
      * timer: DispatcherTimer – časovač pro aktualizaci simulace.
      * isRunning: bool – zda simulace běží nebo ne.
      * frameCount: int, lastFpsUpdate: DateTime – pro výpočet FPS.
    - Metody:
    - Timer\_Tick: Aktualizuje boidy a vykresluje scénu.
    - RenderBoids: Vykresluje boidy jako trojúhelníky na Canvas.
    - StartButton\_Click, PauseButton\_Click, ResetButton\_Click: Ovládají simulaci.
    - ImportButton\_Click, ExportButton\_Click: Spouštějí import/export.

### Boid

* + - Role: Implementuje pravidla chování boida (Separation, Alignment, Cohesion, Boundary Avoidance/Wrap-around).
    - Atributy:
    - Position: Vector, Velocity: Vector – pozice a rychlost boida.
    - MaxSpeed: double, PerceptionRadius: double – parametry pohybu.
    - BoundaryMargin: double, BoundaryStrength: double – konstanty pro vyhýbání se hranicím.
    - Metody:
    - Update: Kombinuje pravidla a aktualizuje pozici/rychlost.
    - CalculateSeparation, CalculateAlignment, CalculateCohesion, CalculateBoundaryAvoidance: Vypočítávají vektory pravidel.

### SimulationState

* + - Role: Datová třída pro ukládání stavu simulace při importu/exportu.
    - Atributy:
    - Boids: List<BoidData> – seznam boidů.
    - SeparationStrength, AlignmentStrength, CohesionStrength, MaxSpeed: double – globální parametry.

### BoidData

* + - Role: Datová třída pro ukládání stavu jednoho boida.
    - Atributy:
    - X, Y, VelocityX, VelocityY: double – pozice a rychlost.

### ImportExport

* + - Role: Zajišťuje import/export stavu simulace do/z JSON souborů.
    - Metody:
    - Export(path, state): Serializuje SimulationState do JSON.
    - Import(path): Deserializuje JSON do SimulationState.

## Testování

Testy jsou navrženy, tak aby byly otestována každá možná situace. První čtyři testy mají společnou část funkčnost Import. Testují tedy různé situace, které mohou nastat. Jsou otestovány v malém měřítku, tak abych mohl říct, jaké bude očekávané chování. U každého z testů tedy popisuji následující:

* Konfiguraci: Jak nastavit simulaci (počet boidů, parametry, počáteční stav).
* Postup testu: Jak spustit a pozorovat simulaci.
* Očekávané chování: Jak by se boidy měly chovat na základě pravidel modelu (Separation, Alignment, Cohesion). Očekávání je založeno na logice modelu, kde vyrovnané parametry vedou k přirozenému hejnovému chování, ale změny parametrů ovlivní dynamiku.

Všechny testovací soubory se nachází v úvodní složce projektu, v podsložce jménem “\_testovani”. Zde jsou soubory pojmenované testN.json, kde “N” označuje číslo testu popsaného níže.

### Test 1: Vyrovnané hejno v trojúhelníku (základní chování)

#### Konfigurace

* Počet boidů: 3 (pozice [100,100], [110,90], [90,110]; rychlosti [1,1], [1,-1], [-1,1]).
* Parametry: Separation = 0.5, Alignment = 0.5, Cohesion = 0.5, MaxSpeed = 2
* Počáteční stav: Boidy v trojúhelníkové formaci, s různými směry, ale blízko sebe.

#### Postup testu

* Importuj JSON soubor test1.json.
* Spusť simulaci (StartButton) a pozoruj 10–20 sekund.

#### Očekávané chování

Boidy by se měly sbíhat do kompaktního hejna díky Cohesion (směřování ke středu), zatímco Alignment vyrovná jejich směry do společného pohybu. Separation zabrání kolizím, takže se nebudou navzájem srážet, ale zůstanou v malé vzdálenosti.

### Test 2: Blízko umístěné boidy (test vyhýbání kolizím)

#### Konfigurace

* Počet boidů: 3 (pozice [100,100], [102,100], [98,100]; rychlosti [1,0], [1,0], [1,0]).
* Parametry: Separation = 1 .5 (vysoké), Alignment = 0.5, Cohesion = 0.5, MaxSpeed = 2
* Počáteční stav: Boidy velmi blízko sebe (skoro se dotýkají), se stejným směrem.

#### Postup testu

* Importuj JSON soubor test2.json.
* Spusť simulaci a pozoruj prvních 5–10 sekund.
* Sniž Separation na 0.2 a opakuj pro srovnání.

#### Očekávané chování

Díky vysokému Separation se boidy rychle rozejdou, aby se vyhnuli kolizím – střední boid zůstane na místě nebo se mírně pohybuje, zatímco okrajové se oddálí v opačných směrech. Alignment a Cohesion je později přitáhnou zpět do volnějšího hejna. Pokud snížím Separation, boidy zůstanou těsněji pohromadě, což může vést k chaotickému pohybu nebo kolizím

### Test 3: Boidy s různými směry (test zarovnání)

#### Konfigurace

* Počet boidů: 3 (pozice [100,100], [110,110], [90,90]; rychlosti [1,1], [1.2,0.8], [0.8,1.2]).
* Parametry: Separation = 0.5, Alignment = 1.5 (vysoké), Cohesion = 0.5, MaxSpeed = 2
* Počáteční stav: Boidy v diagonální formaci s mírně odlišnými rychlostmi.

#### Postup testu

* Importuj JSON soubor test3.json.
* Spusť simulaci a pozoruj rychlosti boidů
* Sniž Alignment na 0.2 a opakuj.

#### Očekávané chování

Vysoké hodnoty Alignment způsobí, že rychlosti boidů se rychle vyrovnají do průměrného směru (přibližně [1,1]), a hejno se bude pohybovat synchronizovaně jako jednotka. Cohesion je udrží blízko, zatímco Separation zabrání přílišnému shluknutí. Pokud snížím Alignment, boidy zůstanou v různých směrech déle, což vede k rozejití hejna

### Test 4: Rozptýlené hejno (test soudržnosti)

#### Konfigurace

* Počet boidů: 3 (pozice [100,100], [200,200], [0,0]; rychlosti [1,0], [0,1], [0,-1]).
* Parametry: Separation = 0.5, Alignment = 0.5, Cohesion = 1.5 (vysoké), MaxSpeed = 2
* Počáteční stav: Boidy rozptýlené po Canvasu s různými směry.

#### Postup testu

* Importuj JSON soubor test4.json.
* Spusť simulaci a pozoruj, jak se boidy sbíhají.
* Sniž Cohesion na 0.2 a opakuj.

#### Očekávané chování

Vysoké Cohesion způsobí, že boidy rychle směřují ke společnému středu hmoty, sbíhají se do těsného hejna a pokračují společně. Separation zabrání přílišnému shluknutí, zatímco Alignment vyrovná směry. Pokud snížím Cohesion, boidy zůstanou rozptýlené a pohyb bude chaotický, bez silného sbíhání.

### Test 5: Náhodné hejno

#### Konfigurace

* Počet boidů: libovolný (pozice náhodné; rychlosti náhodné).
* Parametry: Separation = 1, Alignment = 1, Cohesion = 1, MaxSpeed = 3
* Počáteční stav: Boidy jsou náhodně rozptýlené po Canvasu s různými směry.

#### Postup testu

* Restartuj aplikaci, nebo pouze zastav předešlou simulaci.
* Tlačítkem Reset se předešlá simulace vynuluje a spustí se nová. Tato nová simulace začíná s náhodnými pozicemi a rychlostmi.
* Pozoruj animaci.
* Opakuj postup a sleduj, jak jsou pozice vždy jiné a náhodné.

#### Očekávané chování

Vyrovnané a standartní hodnoty parametrů zajistí normální a očekávané hejnové chování. Tlačítko Reset zajistí randomizovaný start simulace, kdy poté se vše odvíjí právě od této startovní pozice. Každá simulace tedy bude jiná, což se dá vypozorovat právě tímto postupem.

Závěr

ahoj

Zdroje

1. Online. Dostupné z: <https://www.red3d.com/cwr/boids/> [cit. 2025-08-20].
2. Online. Dostupné z: <https://vergenet.net/~conrad/boids/pseudocode.html>. [cit. 2025-08-20].
3. Online. Dostupné z: <https://archive.org/details/boids-demo-reel-1>. [cit. 2025-08-20].
4. Online. Dostupné z: <https://cs.stanford.edu/people/eroberts/courses/soco/projects/2008-09/modeling-natural-systems/boids.html>. [cit. 2025-08-20].