Relazione Progetto Boids

Francesco Bartoli

1 Introduzione

1.1 Scopo

Il programma ha come obiettivo quello di simulare in uno spazio bidimensionale il comportamento di stormi di uccelli in volo, che verranno indicati con il nome di *boids*.

1.2 Installazione

Le istruzioni su come compilare, testare, eseguire sono presentate nel README del progetto, che riporto qui sotto:

Build instructions are for **Ubuntu 22.04**.

1.2.1 Prerequisites

SFML (2.5): Library for graphic representation.
TGUI (1.0): Library for graphic interface.

1.2.2 Clone the Repository

```
git clone https://github.com/Evyal/boids.git
```

1.2.3 Build the Project

Create a build directory

```
mkdir build
cd build
```

Configure CMake in Release mode

```
cmake .. -DCMAKE_BUILD_TYPE=Release
```

Build the project

```
cmake --build .
```

1.2.4 Running the program

```
./boids
```

2 Struttura del programma

Segue una breve descrizione sintetica delle principali scelte progettuali e implementative del programma.

I file del progetto sono organizzati in sottocartelle, nello specifico i .cpp si trovano nella `/source`, mentre i rispettivi header sono situati nella `/include`. La directory `/testing` contiene i file di testing, e infine in `/assets` sono presenti alcuni file necessari per il corretto funzionamento del programma.

2.1 Regole di volo

I *boids* si seguono delle regole di volo, che ne determinano il comportamento. Ad ogni istante, il programma modifica le velocità e le posizioni dei *boids* attraverso le seguenti formule:

$$\vec{v}_{bi} = \vec{v}_{bi} + \vec{v}_S + \vec{v}_A + \vec{v}_C + \vec{v}_R$$

$$\vec{x}_{bi} = \vec{x}_{bi} + \vec{v}_{bi}\Delta t$$

Dove $\vec{v}_S$, $\vec{v}_A$, $\vec{v}_C$, e $\vec{v}_R$ sono rispettivamente:

2.1.1 Separazione

$$\vec{v}_1 = -s\sum_{j\neq i}(\vec{x}_{b_j} - \vec{x}_{b_i}) \quad \text{se} \quad \left|\vec{x}_{b_i} - \vec{x}_{b_j}\right| < d_s$$

2.1.2 Allineamento

$$\vec{v}_2 = a\left(\frac{1}{n-1}\sum_{j\neq i}\vec{v}_{b_j} - \vec{v}_{b_i}\right) \quad \text{se} \quad \left|\vec{x}_{b_i} - \vec{x}_{b_j}\right| < i$$

2.1.3 Coesione

$$\vec{x}_c = \frac{1}{n-1}\sum_{j\neq i}\vec{x}_{b_j} \quad \text{se} \quad \left|\vec{x}_{b_i} - \vec{x}_{b_j}\right| < i$$

$$\vec{v}_3 = c(\vec{x}_c - \vec{x}_{b_j})$$

Dove $n-1$ assume valori dipendenti dal numero di boid nel range di interazione, e non è un valore fisso dipendente dal numero di boids nello stormo.

E s, ds, a, c, i sono parametri della simulazione.

2.1.4 Repulsione

$$\vec{x}_c = \frac{1}{n-1}\sum_{j\neq i}\vec{x}_{b_j} \quad \text{se} \quad \left|\vec{x}_{b_i} - \vec{x}_{b_j}\right| < dr$$

Questa regola determina l'allontanamento tra *boids* di stormi differenti e introduce due nuovi parametri r, dr.

2.1.5 Interazione *On Click*

$$\vec{v}_1 = \pm p\sum_{j}(\vec{x}_{b_j} - \vec{x}) \quad \text{se} \quad \left|\vec{x}_{b_j} - \vec{x}\right| < i$$

Questa regola permette all'utente di interagire con i boids, e introduce l'ultimo parametro che determina il comportamento dei boids. Il $\pm$ è dovuto al fatto che questa interazione può essere sia attrattiva che repulsiva mentre $\vec{x}$ è il punto in cui l'utente ha cliccato.

2.2 File di implementazione

Tutti i file del progetto sono stati inseriti in un namespace. Tutti i file menzionati di seguito come .cpp hanno un corrispettivo header.

2.2.1 constants.hpp

Namespace che contiene valori come limiti di velocità o posizione per i *boids*, parametri di interazione di default, o ulteriori valori per l'inizializzazione degli elementi dell'interfaccia grafica.

2.2.2 structs.hpp

Struct per impacchettare dei valori usati per inizializzare bottoni o altri elementi di interfaccia grafica

2.2.3 boid.cpp

File di implementazione della classe Boid e funzioni ausiliare per gestirne il comportamento

2.2.4 flock.cpp

File di implementazione della classe Flock che determina la struttura collettiva dei *boids* all'interno di uno stormo.

2.2.5 random.cpp

File che si occupa della generazione di numeri casuali

2.2.6 statistics.cpp

File che si occupa del calcolo delle statistiche restituite a schermo, riguardanti valori medi e deviazioni standard delle posizioni e velocità dei *boids*.

2.2.7 graphics.cpp

Breve file che contiene una funzione per la corretta rappresentazione grafica dei *boids*, ed un'altra che costruisce un rettangolo (sf::Rectangle) prendendo come parametro una delle struct definita nel file sopracitato.

2.2.8 switchbutton.cpp

La classe Switchbutton introduce la funzionalità di un bottone che può trovarsi in due stati, non fornita da TGUI.

2.2.9 gui.cpp

Classe che introduce gli elementi necessari a costruire l'interfaccia grafica del programma, e si occupa di coordinare tutti i file che implementano la logica all'interno di essa.

3 Interfaccia della simulazione

4 Testing

Tutti i file incaricati dell'implementazione di parte della logica del programma hanno un corrispettivo file di testing. Più precisamente, sono presenti i seguenti: testboid.cpp, testflock.cpp, testrandom.cpp e teststatistics.cpp.

Attraverso i test si è cercato di controllare che le classi, i metodi delle classi e le funzioni introdotte fossero esenti da errori e mostrassero il comportamento atteso. Sono stati eseguiti test in casi semplici per poter stabilire il funzionamento corretto del codice, e anche in alcuni casi particolari quando ritenuto necessario.

Il framework che si è utilizzato per creare le testing unit è doctest.h, il cui file è incluso nella cartella nel progetto (/assets/doctest.h). Questa libreria è in grado di generare autonomamente un main e permette l'esecuzione dei test semplicemente includendo il file sopracitato.

Per potere eseguire i test è necessario trovarsi nella cartella dove vengono prodotti gli eseguibili dei file precedentemente menzionati, seguendo i passaggi elencati sotto. Immaginando di trovarsi nella cartella principale dov'é contenuto il progetto:

```
    cd build
    cd testing
```

E digitare il comando corrispondente al test che si vuole eseguire:

```
    ./testboid
    ./testflock
    ./testrandom
    ./teststatistics
```

5 Conclusioni

5.1 Descrizione dei risultati