Dokumentacja wstępna projektu na przedmiot ZPR dla dr inż. Rafała Biedrzyckiego

Projekt: Gra w Życie

Autorzy:

Damian Mazurkiewicz

Radosław Nowak

Rozwinięty opis zadania:

Zadaniem jest zasymulowanie bytowania dwóch populacji (mięsożerców i roślinożerców) i wyświetlanie wyników symulacji w czasie rzeczywistym. Poniżej wypisane są zasady tej symulacji:

- Początkowe warunki symulacji (ilość roślinożerców, mięsożerców, ich położenie na mapie i cechy) będą losowe.
- By przetrwać, osobniki muszą jeść, niedobór pożywienia skutkuje śmiercią. Mogą także umrzeć śmiercią naturalną, na skutek choroby, lub zostać zabite.
- Pożywieniem roślinożerców jest trawa, która występuje na całej planszy, więc mogą one zaspokoić swój głód w każdym miejscu. Jednak podczas jedzenia znacząco zmniejsza się ich zasięg i pole widzenia i mogą łatwiej paść ofiarą mięsożercy. Pożywieniem tego drugiego są głównie roślinożercy (choć mogą też wystąpić akty kanibalizmu).
- Stworzenia mogą się mieć potomstwo z osobnikami innej płci. Dziecko dwóch osobników odziedziczy ich współczynniki zgodnie z odpowiednim algorytmem dziedziczenia (uwzględniającym mutacje). Potomstwo pojawia się od razu, jednak musi przejść okres dzieciństwa, w którym jest ono osłabione (a w przypadku mięsożerców nie może nawet polować i jedzenie muszą dostarczyć im rodzice).
- Osobniki mają unikalne dla siebie priorytety zachowań. Weźmy na przykład pewnego roślinożercę, którego zachowania uszeregowane od najważniejszego do najmniej ważnego to : jedzenie, bezpieczeństwo (staranie się przebywać jak najdalej od drapieżników), sen, wydanie potomstwa. Wtedy gdy wszystkie potrzeby ma zaspokojone, to będzie jadł. Jednak gdy będzie bardzo zmęczony, ale najedzony, to pójdzie spać mimo niskiego priorytetu tej czynności.
- Każdy przedstawiciel populacji ma przydzielony zespół współczynników, jednak by nie doprowadzić do sytuacji, w której każdy u każdego osobnika wyewoluują maksymalne

wartości atrybutów, są one zależne od siebie i przydzielanie na zasadzie „coś za coś”. Oto lista cech i zależności między nimi:

- Siła – opisuje skuteczność ataku dla mięsożerców i skuteczność obrony roślinożerców. Na przykład jak przedstawiciel roślinożerców zostanie zaatakowany przez mięsożercę o niższym współczynniku siły, to się obroni i zabije polującego. Kosztem który ponosi się za większą siłę jest większy apetyt.
- Szybkość – odpowiada za szybkość poruszania się osobnika, jednak im jest większa tym ze bardziej się on męczy i potrzebuje więcej snu.
- Zasięg widzenia – odpowiada za to, z jakiej odległości stworzenie jest w stanie dostrzec swojego wroga/ofiarę. Za duży zasięg widzenia płaci się węższym polem widzenia.
- Płodność – odpowiada za ilość potomstwa jaką może wydać przedstawiciel danej populacji. Jednak dużą płodność okupuje się krótszym życiem.

- Na współczynniki i zachowania osobnika wpływają także jego cechy i stany. Oto ich lista.
  - Dzieciństwo – występuje w początkowej fazie życia stworzenia, wtedy ma ono obniżoną siłę. Drapieżniki ponadto w tym okresie nie mogą polować (jedzenie muszą zagwarantować mu rodzice).
  - Choroba - może być nabyta przez osobnika zupełnie przypadkowo w trakcie jego życia lub odziedziczona. Obniża ona współczynniki osobnika.
  - Agresja - powoduje, że osobnik może atakować też przedstawicieli swojego gatunku.

Funkcjonalności programu:

- Podglądanie wyników symulacji w czasie rzeczywistym w środowisku graficznym (widok z lotu ptaka). Zostanie wykorzystana biblioteka SDL 2.
- Możliwość „przyjrzenia się” dowolnemu osobnikowi. Polega ona na tym, że po kliknięciu na osobnika, z boku wyświetli się lista jego cech, płeć, jego stan, aktualne zajęcie , pokazane zostanie także jego pole widzenia.
- Możliwość przesuwania kamery za pomocą strzałek na klawiaturze. Obszar symulacji będzie większy niż obszar widziany na ekranie.

Lista zadań i oszacowanie ich pracochłonności:

Damian Mazurkiewicz	Radosław Nowak
● Klasa View ○ Metoda drawCreature(args) - 2h ○ metoda drawCreatureInfo(args) - ■ rysowanie pola widzenia - 0.5 h ■ rysowanie listy atrybutów i cech 1h ○ metoda drawBackground() - 1h ○ obiekt, który ułatwia pracę z biblioteką SDL - 3h ○ metoda run() - zawierająca główną pętlę programu, ta metoda będzie wyświetlała symulację na ekranie, jednocześnie będzie przesyłała do kontrolera informacje o zdarzeniach takich jak (wciśnięto strzałkę na klawiaturze, kliknięto myszką, minął kwant czasu w grze[co wymaga aktualizacji statystyk osobników]. - 2h ○ tworzenie wyjątków i ich obsługa - 2h. ● Klasa Controller ○ komunikacja z klasą Model - 2h ○ obsługa poruszającej się kamery -2h ○ obsługa kliknięcia na stworzenie - 1h.	● Klasa Model - Metoda createHerbivore(...) - 0,5h - Metoda createCarniivore(...) - 0,5h - Metoda updateAnimalsStatuses(...) - 1,5h - Metoda getAnimalsInTriangle(...) - 1h - Metoda getCoordinates(...) - 0,5h ● Klasa czysto wirtualna Animal - 3h ● Klasy Carnivore i Herbivore dziedziczące po klasie Animal - Metody updateStatus(...) - 4h - Metoda, która aktualizuje status jednostki, taki jak np. czy jednostka żyje, czy zachorowała etc. Wywoływana co określony kwant czasu, a nie w każdej klatce. - Metody doMove(...) - 2h - Metoda wywoływana w każdej klatce, aktualizuje położenie obiektu, jego prędkość etc. - Metody zwracające dane dotyczące konkretnej jednostki - 1h ● Klasa Traits - zawiera zbiór cech każdej jednostki. 0,5h ● Klasa Attributes dziedzicząca po Traits. Zawiera Atrybuty danej jednostki wraz z interfejsem - inheritAttributes(...) - Metoda implementująca dziedziczenie cech po przodkach. 2h - updateAttributes(...) - Metoda, która na podstawie danego statusu jednostki aktualizuje jej tymczasowe atrybuty. Np chora jednostka jest słabsza, a słaba wolniej biega. 2h ● Dokumentacja 2h
Suma godzin: 16,5	Suma godzin: 20,5