448.057 Context-Aware Computing (UE)
Exercise 3 – School of Fishes escaping Predators

Task description

The third system to be modelled using NetLogo consists of a school of fishes escaping one or more predators. As many living beings, also fishes tend to cluster in groups and adapt their movements to the ones of the group. This schooling behaviour increases the efficiency in hunting food and raises the probability of survival from a predator's attacks. In this exercise, different adaptation strategies and their impact on the survival of fishes in the long term will be investigated.

Fishes and predators (in this specific assignment we assume the latter are sharks) have to be modelled as different breeds. The number of fishes and sharks is controllable by the user through the input variables `total-fishes` (range $[1-100]$) and `total-sharks` (range $[0-20]$). Sharks and fishes are randomly dispersed in the virtual world, which is a piece of ocean (blue shade) modelled as 40x40 torus that wraps horizontally and vertically. Sharks and fishes swim in the ocean with a given speed. The latter is specified by the input variables `movements-sharks` (range $[1-6]$) and `movements-fishes` (range $[1-6]$), i.e., these variables specify how many patches a fish or a shark moves every tick. Furthermore, sharks and fishes also have specific ranges of sight, and their values are specified by the input variables `sight-sharks` (range $[1-10]$) and `sight-fishes` (range $[1-10]$).

Sharks eat fishes to survive, so they move around to hunt fishes. If they identify one or more fishes in their range of sight, they will head towards one of them (picked up randomly), whereas if they do not see any fish in their surroundings, they just move in a random direction. Each shark has an amount of energy, and when a shark moves of one patch, it consumes `sharks-energy-expenditure` amounts of energy, where the latter is a variable specified by the user in the range $[1-10]$. When a shark eats a fish, it acquires some energy. A fish eaten corresponds to `sharks-acquired-energy` units of energy. Also the `sharks-acquired-energy` variable is specified by the user and its range is $[1-100]$. In the beginning, sharks have an initial amount of energy equal to `sharks-initial-energy` (taken from input through a slider with range $[1-10000]$). The energy can be lost only by moving around, and when a shark does not have any more energy (zero or below), it dies. For simplicity, fishes are assumed to live until they are eaten by sharks and never get tired. No new fish should be generated once the simulation has started. Therefore, when no fishes remain in the field, all sharks will die. On the contrary, when all sharks starved, the remaining fishes will live forever.

Fishes move according to simple rules. If there is no shark in the surroundings, a fish follows the heading of the other fishes in its neighbourhood, if any; otherwise it simply moves randomly. If, instead, one or more sharks are in the range of sight, the fish will have two options. If there are no other fishes in the surroundings, i.e., only one or more sharks are detected in the range of sight of the fish, the fish will escape in the opposite direction with respect to the position of one of the sharks (i.e., it heads towards a shark and rotates 180 degrees). If other fishes are present in the surroundings, the fish will follow a direction $H = (\alpha \cdot S) + ((1-\alpha) \cdot F)$, where:

- α is a variable with range $[0, 1]$ taken from input;
- S is the heading opposite to the shark's position;
- F is the average heading of the other fishes in the surroundings.

Michael Stocker, Robert Thomann <michael.stocker@tugraz.at, r.thomann@tugraz.at>

The rationale behind this is as follows: by swimming into direction S, a fish may escape the shark on the one hand, but on the other hand this may result in breaking up the swarm, making it easier for the shark to catch isolated fishes later. By following direction F, the fish stays with the swarm ignoring the presence of the shark (the swarm will not change its original direction even in presence of predators). The above approach represents a compromise between these two extremes where $(\alpha \cdot S)$ can be set to any value between 0 (fish will follow F) and 1 (fish will follow S). This scenario, that does not exactly represents the behaviour of fishes in reality, has to be modelled and simulated using NetLogo. The simulation should contain two buttons: `setup` and `go`, associated to the `setup` and `go` procedures, respectively. The `setup` procedure creates the ocean and generates an initial amount of fishes and sharks spread randomly. These operations are executed inside the procedures `setup-ocean`, `setup-fishes` and `setup-sharks` respectively, as shown in Listing 1.

Listing 1: Skeleton for the `setup` procedure.

```
to setup
        clear-all
        reset-ticks
        setup-ocean
        setup-fishes
        setup-sharks
end
```

The `go` procedure models the daily activities ongoing in the ocean: sharks will hunt fishes, and fishes will try to escape from sharks, as detailed above. The go procedure will thus consist of the procedures `move-fishes` and `move-sharks`, as shown in Listing 2.

Listing 2: Skeleton for the `go` procedure.

```
to go
        ask fishes [ move-fishes ]
        ask sharks [ move-sharks ]
        tick
end
```

Basic functionality: flash expansion *(6 points)*. Create a NetLogo model that follows the specifications above and make sure that the program does not enter an infinite loop. Make also sure the school of fishes generate a flash expansion when a shark is trying to attack them. In order to have a clearer view of the ongoing activities in the ocean, three NetLogo monitors have to be created as well: two showing the amount of fishes and sharks left, and one showing the average energy of sharks.

System behaviour *(3 points)*. Discuss the high-level behaviour of the system.

1. What is the impact of the different input variables of the model? Please respect the space constraints.

Michael Stocker, Robert Thomann <michael.stocker@tugraz.at, r.thomann@tugraz.at>

Answer:

The movements of sharks and fish don't matter if the sight of sharks is high, all fish die and all sharks survive in such a case.
If the sight of sharks is low, they usually all die, even if their movements is much higher than the movements of fish. Also changing the energy variables doesn't change much in this case.

2. Assess the role of the variable α. Understand the characteristics of the function $Y = f(X)$, where Y is the number of fishes surviving in the long term (after the shark died), and X is the variable α in the range $[0, 1]$. To this end, the students should run the simulation with different α values. For each α value, multiple runs should be performed and the results should be averaged. Please write down your results and conclusions, and specify the parameter configuration(s) you have used.

Answer

Configuration:
Total-fishes 50, total-sharks 3, movements-fishes 1, movements-sharks 1, sight-fishes 5,sight-sharks 10, initial E 5221, E expenditure 5, acquired E 15

Results:
alpha = 0.0 -> Y = 29
aplha = 0.4 -> Y = 31.2
aplha = 0.8 -> Y = 45.3
aplha = 1.0 -> Y = 47.6

Conclusion: We can see that for this setting the higher the alpha is, there is the higher number of survived fish. So it is better to leave the school in case of shark attack, at least for this configuration.

Extended functionality: fountain effect *(1 point)*. Update your basic NetLogo model so that the school of fishes generates a fountain effect when attacked by a shark. When solving this functionality you are allowed to relax some of the above specifications, e.g., you can assume that the shark does not continuously change direction randomly.

Please submit the filled PDF and your NetLogo source code to the TeachCenter as illustrated in the lecture within **Sunday, 9.12.2018, 23:59 CET**. Please submit the extended solutions as separate files or use a switch to enable or disable an extended functionality.

Group ID: 13

Stud. Names: Juraj Vida
Timur Cerimagic

Finalize Form