Henry Nguyen

December 15, 2015

CSCI 4210

Term Project Report

For the term project, I chose to use Pickle to simulate a flock of sheep. Pickle is an agent-based modeling and simulation platform. Everything in a Pickle simulation is serializable as an xml file. This makes it easier to store, transfer, and generate simulations. Pickle uses behavior-based controllers for its agent logic, which provides a powerful set of tools for inscribing intelligent agents and dynamic environments. Lastly, Pickle is designed to be able to run on multiple simulation kernels with many visualizations options. For this sheep simulation, MASON is used as the simulation environment.

The main file of the simulation is an xml file called SheepFlocking.xml. In this file, Agents are declared and agent functions are called. The whole application is called SheepFlocking. In this application, both the driver and graphical user interface rely on MASON to provide the simulation environment. The simulation window size is declared at 640x480 however, the simulation world is 1280x720. The friction parameter is set at 0, the lowest value, to denote no movement friction in the simulated world. The first agent that is created is the Sheep agent. This is the main agent of the simulation used to simulate their flocking behavior. In this simulation, there are 40 sheep agents depicted by the color blue. The sheep agents are initially set at a size and max speed of 10 and a turning radius of 30 degrees. The 10 used in the parameters are trivial values used to

initialize default attributes to the first agent. Other agents are created with attributes relative to the default attributes of the sheep agents. Sheep are very social animals and tend to stay together in groups of 5 or more. Sheep that are not in a group tend to stray away from the flock to get lost or eaten by predators. A sheep agent will move around to find another sheep agent. Once a sheep agent has found another sheep agent, they will move towards each other until they get too close and move away until they get too far and repeat. This is denoted by the controller functions getNearestSheepInner and getNearestSheepOuter. These two functions may seem as if they're counterintuitive when it comes to flocking, but the functions prevent the sheep agents from clumping together. When the agents are in a group, the sheep will mimic each other's movement and direction and thus create the flock's movement. The movement mimicking is created by the controller function getNearestSheepCollide. Sheep that are not in a flock or have strayed away will continue to move around by themselves until they find another sheep agent or a flock to mimic.

When it comes to sheep and lambs, predators may include mountain lions, foxes, coyotes, bobcats, and more. For simplicity, only the coyote agent is used. In this simulation, there are 4 coyote agents, whose size is slightly larger at 15 and max speed is slower at 8. These agents are depicted with a red color. The coyote agents move independently and their main purpose is to find and chase after the nearest sheep. The coyote agents will move randomly until it finds a sheep in its radius and chase after it. Once a sheep agent is chased down, the coyote agent will initiate the chomp function, which removes the sheep from the simulation to depict it being eaten. Coyotes are usually

lonesome predators and hunt alone therefore the coyote agents will move away from each other to ensure that their prey does not have to be shared.

Dog breeds such as Anatolian shepherds and Great Pyrenees are considered good breeds to be livestock protection dogs. The main purpose of these guardian dogs is to chase away predators. They differ from herding dogs, which are dogs used to manage the flocks of sheep. Herding dogs will be another agent of the simulation discussed later in this report. Both guardian and herding dogs are raised together with the sheep to create a bond between the animals. The bond is helpful to ensure that the sheep are not frightened by another animal as they usually can be. There are two guard dog agents in this simulation, and they are slightly larger than the coyotes at 16 with a speed of 9. With a real sheep flock, a shepherd will breed the guard dogs to have a color similar to the wool of the sheep. The color will be less of a threat for sheep and will make identification by predators more difficult. However, in this simulation, the guard dogs are green. The main purpose of the guard dogs in the simulation is to chase away the coyotes. Similar to the coyote agents chasing the sheep agents, once a guard dog agent chases down a coyote agent, the coyote agent will disappear to simulate it being chased away. The guard dog agents will also follow the sheep agents around until it detects a coyote agent in its radius, in which it will then chase after the coyote agent. The sheep agents, however, will not follow the guard dog agents because the guard dogs are usually chasing after a coyote and the sheep do not want to get near a coyote. The guard dog agents share the same getNearestSheep function as the coyote agents.

Popular breeds for herding dogs include border collies, Australian shepherds, and others, but most shepherds will use border collies due to their intelligence. In this

simulation, there will also be two herd dog agents. The agents are black in color with 11 as the size and 12 for the speed. Herd dogs are faster as they usually run ahead of the flock to influence movement and direction. The main purpose of the herd dog agents is to influence flocking behavior so they will not follow the sheep agents, but instead the sheep agents will follow the herd dog agents. The herd dogs do, however, follow each other to bring two flocks of sheep together as a one big flock. Similar to the sheep agents, when the herd dog agents come together, they will mimic each other's movements instead of clumping together. Their mimicking function is called getNearestHerdDogCollide.

Each agent in the SheepFlocking application has a controller file, which defines each function for the agents. The sheep agents have SheepController.xml as their controller file, and this file has the function definitions along with the priorities in which each function hold. With 1 being the lowest priority, getRandomPoint is the first function in each of the agents' controller files. The getRandomPoint function will periodically be called and cause the agents to change directions to a random point if they have no movement direction to begin with. Any sheep agent that strays away from a flock will randomly move in any direction until it finds a flock. The functions getNearestSheepInner and getNearestSheepOuter are prioritized at 2 and 3, respectively. As mentioned before, these two functions keep the sheep agents together but not as to restrict movement. Because sheep agents also follow the herd dog agents, getNearestDogOuter and getNearestDogInner are prioritized at 4 and 5, respectively. These functions are higher than the sheep functions because the sheep agents should follow the herd dog agents instead of the sheep agents if the herd dogs are present. The

function getNearestSheepCollide was mentioned before, and it has a priority of 6 in order to get the sheep agents to flock together. At the priority of 7, getNearestCoyote is the function that causes the sheep agents to flee from the coyote agents. At the highest priority for all agents is the getNearestEdge function, which prevents the agents from clumping together at the edge of the simulation world. Each of these functions has a motor schema type of attraction, repulsion, or mimic. A schema of attraction will attract the agent to the target, repulsion will cause the agents to move away, and mimic will cause the agents to mimic the target's movement and direction.

In CoyoteController.xml, the function getNearestCoyote has a schema type of repulsion since coyotes tend to hunt alone. Above that priority is the getNearestSheep function, which causes the coyote agents to chase after the sheep agents. However, the coyote agents would rather avoid the guard dog agents than get ahold of a sheep, so the getNearestGuardDog function has a higher priority than the chasing sheep function.

In GuardDogController.xml, the priorities are reversed. The guard dog agents will follow the sheep agents until a coyote agent comes close to it, then it will chase after the coyote. In HerdDogController.xml, the herd dog agents will move around and try to look for each other and then mimic each other's movement behaviors. The herd dog agents will not follow the sheep agents because they are trying to influence the sheep agent behaviors and not the other way around.

The simulation starts with every agent at random positions, and once the simulation runs the agents will proceed to carry out their functions. The coyote agents will try to eat as many sheep agents as possible before the guard dog agents come and

chase them away. Once all four coyote agents are chased away, the simulation continues to run until it is stopped.

In order to run the Pickle Simulation, sbt (Build tools for scala) must be on the system as mentioned in the README provided by the creator of Pickle, Terrance Medina. All other files that came with the Pickle Simulation remain the same. The two tutorial videos provided by Medina were also used as resources in creating SheepFlocking.xml and the other files associated with it.