TPBoids_Sheeps

Objectif

L'objectif de cet exercice est d'implémenter l'algorithme décrit dans l'article suivant : http://www.csc.kth.se/utbildning/kth/kurser/DD143X/dkand13/Group9Petter/report/Martin.Barksten.David.Rydberg.report.pdf

Cet article propose une variante de l'algorithme de Reynolds sur le comportement d'une nuée d'oiseau ou d'un troupeau. Le but est d'adapter cet algorithme à un troupeau de mouton.

1 - Berger

1. Creer un script ***KeyboardControl***.
2. Ce script doit permettre de déplacer l'objet auquel il est attaché à l'aide du clavier.
3. Appliquer ce script au berger.

2 - Chien

1. Creer un script ***MouseControl***.
2. Ce script doit permettre de déplacer l'objet à l'aide d'un clique de souris. L'objet doit se déplacer graduelement vers l'emplacement cliqué par la souris.
    i. Détecter un clique gauche de la souris
    ii. Envoyer un raycast depuis la position du curseur à l'aide de la fonction *Camera.ScreenPointToRay()*.
    iii. Récupérer la position touchée dans une variable *target*
    iv. Déplacer l'objet graduelement vers *target*.
3. Appliquer ce script au chien.

3 - Moutons

Dans le script ***SheepBoid*** :

1. Implémenter la fonction sigmoid 3.1.
    - Changer la valeur du dénominateur **20** par **0.3**.
2. Implémenter la fonction de poids 3.2.
3. Implémenter la fonction inverse 3.3.

4. Implémenter la régle de cohésion 3.4.
    - Une liste de tout les moutons est accessible via la variable *SheepHerd.Instance.sheeps*.
    - Bonus : Visualiser le vecteur produit à l'aide d'un *Debug.DrawRay()* de couleur vert.
5. Implémenter la régle de séparation 3.5.
    - Bonus : Visualiser le vecteur produit à l'aide d'un *Debug.DrawRay()* de couleur noir.
6. Implémenter la régle d'alignement 3.6.
    - Remplacer la notion de rayon de **50 pixels** par une variable ***alignementZoneRadius*** (valeur en annexe).
    - Bonus : Visualiser le vecteur produit à l'aide d'un *Debug.DrawRay()* de couleur jaune.
7. Implémenter la régle de fuite 3.8.
    - Changer la valeur du paramètre **10** par **2**.
    - Le predateur est accessible dans la variable *predator*.
    - Bonus : Visualiser le vecteur produit à l'aide d'un *Debug.DrawRay()* de couleur rouge.
8. Implémenter la formule 3.9 dans la fonction *Vector3 ApplyRules()*.
    - Bonus : Reprendre les *Debug.DrawRay()* précédents pour leur appliquer les poids de chaque règle.
9. Ajouter une régle d'encloisonement, pour faire en sorte que les moutons restent dans l'enclos.
    - Cette règle est déjà préfaite sous la forme de la fonction *Pen.Instance.RuleEnclosed()*.
    - Donner un poids de **3** à cette règle.

4 - Bonus

1. Proposer une règle supplémentaire qui permette aux moutons d'interagir avec le berger, par exemple une régle d'évitement pour éviter les collisions avec le berger, ou encore une régle de cible, pour permettre aux moutons de suivre le berger.

Annexe - Valeurs des variables

Variable	Value
flightZoneRadius	7
weightCohesionBase	0.5
weightCohesionFear	5

Variable	Value
weightSeparationBase	2
weightSeparationFear	0
alignementZoneRadius	3
weightAlignmentBase	0.1
weightAlignmentFear	1
weightEscape	6