Ist es möglich eine realistische Simulation eines, durch Anwender-Interaktion beeinflussbares, Schwarmverhaltens in der Unity-Engine performant zu gestalten?

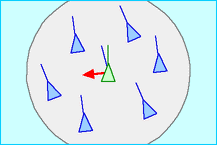
1.1 Themen Definition

In dieser Arbeit werden die drei Grundlegenden "Steering Behaviours" begriffe von Craig Reynold beschrieben, "Separation", "Alignment" und "Cohesion", um eine natürliche Animation von Tierschwärmen in Unity zu simulieren. Anschließend wird in der Unity-Engine getestet ob dies Performant abläuft unter Anwender-Interaktion, desweiteren als Impact benannt.

2. 1 Grundlagen des Flockings

Schwarmverhalten beschreiben die Bewegung eines einzelnen Individuums, auch Boid oder autonomer Agent genannt, innerhalb eines Schwarms mehrerer Boids, sowie das daraus resultierende Verhalten aller Boids als Schwarm. (Buckland, 2005)

2.2 Alignment



(Abbildung 1, Alignment) \*

Das Alignment ist dafür zuständig, dass die Boids sich in eine vorgegebene Richtung und Geschwindigkeit Orientieren und somit die Positionsänderung vorgenommen wird.

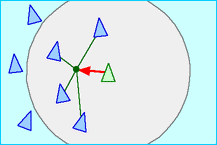
2.3 Separation



(Abbildung 2, Separation) \*

Die Separation sorgt dafür, dass die Boids untereinander einen vorgegebenen Abstand einhalten. Dadurch kann die Separation genutzt werden eine Kollisionserkennung nicht notwendig zu machen, um so Performance zu sparen.

2.4 Cohesion



(Abbildung 3, Cohesion) \*

Die Cohesion wird verwendet, um zu verhindern dass die Boids sich zu weit voneinander entfernen sodass der Schwarm keine Boids verliert. Die Verbindung von Cohesion und Separation wirkt sich somit auf den Abstand zwischen den Boids aus und die Geschwindigkeit der Distanz Veränderung.

2.5 Impact

(Abbildung 4 hier selbst erstellen)

(Berechnung für die Interaktion ("move away from target"))

3 Methodik

-Bevor die Performance getestet werden kann, muss geguckt werden wie ein Tierschwarm auf eine Interaktion Reagieren würde. Hierfür wurde der Vergleich mit einem Fischschwarm gemacht, und wie in Abbildung zu sehen der Impact erstellt.

3.1 Engine Auswahl

- Er wird die Unity Engine (Version 5.3.2) benutz, da:

- hier viele Hilfsmittel schon parat sind. Beispielsweiße Mathe Klassen und das NavMesh

-Unity ist Kostenlos

-Desweiteren bietet Unity eine gute Dokumentation

3.2 Programmiersprache

-Da die Unity Engine genutzt wird, wird als Programmiersprache C# benutzt

-C# wird fortlaufend Weiterentwickelt

3.3 Entwicklungsumgebung

-Als Entwicklungsumgebung wird microsoft visual studio 2015 community edition verwendet

3.4 Versionsverwaltung

-Es wurde ein Privates Repository auf GitHub in Kombination mit dem SourceTree erstellt da das Projekt auf verschiedenen Rechnern nicht nur synchronisiert, sondern auch getestet wird.

4 Durchführung

-NavMesh für die bewegung verwenden

-Impact passiert

-Vector2i Rechnung passiert für den Impact

-Optischer Test wird durch geführt um zu überprüfen ob Algorithmus stimmt

4.1 Praxis Test

5 Ergebnisse

* 1. Auswertung
  2. Tabellen & Diagramme

5.3 Praxis Test

6 Zusammenfassung

6.1 Fazit

Literaturverzeichnis:

Abbildungsverzeichnis:

Abbildung 1, Abbildung 2, Abbildung 3: http://www.red3d.com/cwr/boids/ by [Craig Reynolds](http://www.red3d.com/cwr/index.html)

Abbildung 4 ist selbstgemacht