Pigisland – KMINT – Rob Biemans 2108369

# A\*

## Gekozen afstandsfunctie

De afstandsfunctie kijkt hemelsbreed met behulp van gekozen heuristiek richting het doel vanuit de huidige knoop en berekent de afstand per omliggende knoop naar het doel, om zo uiteindelijk te bepalen welke route (aan knopen) mogelijk dichterbij het doel brengt. Iedere keer wordt de best berekende route (aan knopen) opgeslagen en vervolgens vergeleken met de volgende mogelijke kortste route in een priority que (priority is gebaseerd op ‘movement cost’)

## Heuristiek

Er is gekozen voor de manhatten distance, omdat het speelveld bestaat uit vierkante map waarbij iedere knoop keuze heeft uit 4 edges (niet mogelijk om diagonaal te bewegen). Het is admissable doordat de lengte waarin gekeken wordt altijd gelijk is aan de afstand en evt cost/weights tussen de huidige knoop en het doel (ook een knoop) en dus niet buiten de lengte daarvan en richtingsgevoel geeft aan de afstandsfunctie.

Stel dat lengte 10 (knopen) is vanaf huidige knoop (oorsprong), waarvan 1 een cost/weight heeft van 4 (rotsen) dan is het 10 + 4 = 14 knopen. Binnen een lengte van 14 knopen (horizontaal of verticaal) wordt de route berekend.

## Zwaarte van Edges

Met behulp van Dijkstra in de A\* wordt er rekening gehouden met de ‘movement cost/weight’ factor per knoop om naar het doel te gaan (aangezien rotsen een hogere ‘cost’ heeft van 4, zal indien het beter is erom heen een route kiezen.)

## Strategie van Knabbel en Porcus Vincit

Zowel Knabbel als Porcus Vincit zal enkel bij Enter() van de desbetreffende state (Hunt/Repair) de A\* gebruiken om eenmalig een route te calculeren naar zijn eindbestemming (Varken bij Knabbel / aanmeren voor Onderhoud bij Porcus Vincit) Bij beide states zal tijdens de Execute() de route worden afgelegd en bij Exit() de route untaggen.

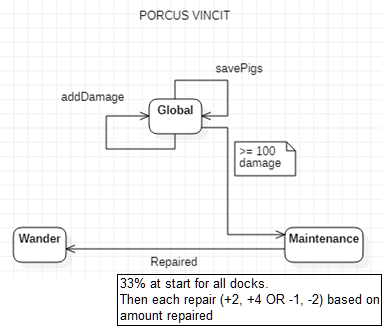
Knabbel gaat in Hunt state als het een varkentje ruikt en de Porcus Vincit zal pas in Repair state gaan, wanner die voor 100 punten is beschadigd. Meer details over de states Enter(), Execute() en Exit() vind je in de volgende paragaaf.

# Finite State Machine

## Toestandsdiagrammen

**Algemeen:** In de Enter(), Execute() en Exit() functie van iedere state is het logica (zoals condities) van desbetreffende state verwerkt, hieronder toont dus een abstractie versie.

**Porcus Vincit**

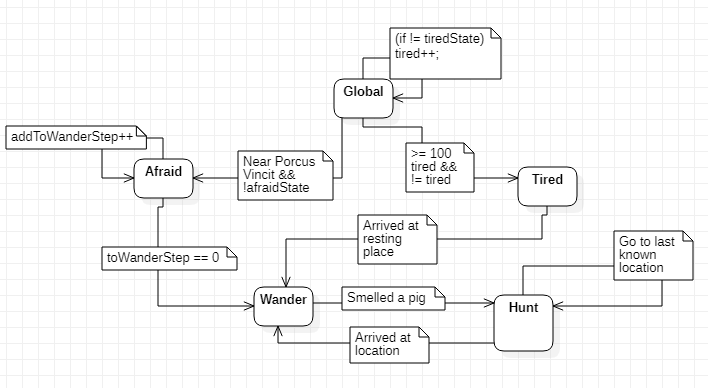


## Aanpassingsstrategieën van Porcus Vincit

Porcus Vincit krijgt om te evalueren waar je de beste kans hebt om vaak voor het meest te kunnen worden gemaakt. In het begin zal iedere aanmeerplek een gelijke kans hebben en hoe langer het spel vordert, des te meer de kansen zullen wijzigen tot een maximum van 80% en als minimum 10%. Porcus Vincit evalueert zich iedere keer bij het aanmeren.

Desbetreffende aanmeerplek zal een verhoogde kans krijgen die gekozen wordt door Porcus Vincit, wanneer het voor meer dan 30 schadepunten wordt gemaakt zal de kans van de desbetreffende aanmeerplek met 2% verhogen als beloning en derden verlagen met 1% als straf. Indien meer dan 50 schadepunten wordt gemaakt dan zal de kans met 4% verhogen en derden verlagen met 2%.

**Knabbel**



# Force-driven entities

## Krachten

**Wander**

Een varkentje zal een willekeurige locatie uitkiezen om daar vervolgens in de buurt binnen een bepaalde straal vandaan stap voor stap heen te bewegen.

**Seek**

Is een kracht die gebaseerd is op het doel van het varkentje (in dit geval Porcus Vincit) waarbij rekening wordt gehouden met het gewicht van het varkentje, afstand tot het doel en maximum snelheid

**Flee**

Is het omgekeerde van vorige kracht Seek. Flee houdt daarbij wel rekening mee met ‘paniek modus’ wanneer een varkentje te dicht bij Knabbel komt.

**Wall Avoidance**

Deze kracht zorgt er voordat de varkentjes niet op bepaalde eilandjes of buiten het scherm kunnen komen. Er wordt een onzichtbare muur getrokken tussen twee punten op verschillende plekken, waarbij gekeken wordt of een muur dicht bij het varkentje komt en op welke punt. Vervolgens wordt er een kracht toegepast om het varkentje te afremmen en van de muur vandaan te bewegen.

**Flocking (Cohesion, Separation, Alignment)**

Flocking is een groepsgedrag (kracht) gebaseerd op cohesion, separation en alignment (richting). Ten eerste worden omliggende varkentjes getagged als ‘neighbour’ binnen een straal waar samen het flocking effect van toepassing is.

Separation zorgt voor een kracht waarbij omliggende varkentjes (oftwel ‘neighbours’) de hoeveelheid uitspreid van elkaar als groep.

Alignment zorgt ervoor dat een groep dezelfde richting ‘met de neus’ gaat.

Cohesion zorgt er weer voor dat de groep niet te ver uit elkaar gaat, en blijft daardoor gevoelsmatig in de buurt van het ‘centrum/massa’ van een groep

## Strategie

De krachten worden eerst individueel berekend en vervolgens bij elkaar toegepast met de weight vanuit het varkentje chromosoom zelf (die krachten vermenigvuldigd) en om het spel ervaring te bevorderen is er ook nog gekozen voor een hardcoded weight. De uiteindelijke berekende kracht zal worden gecontroleerd op maximale beweging/kracht en vervolgens uitgevoerd worden.

# Genetische Algoritme

1. **Populatie**

Er wordt gekeken naar aantal overgebleven varkentjes die gered zijn door Porcus Vincit en daarop gebaseerd nakomelingen gemaakt. Indien er geen varkentjes zijn, dan een nieuwe generatie.

1. **Fitness van Chromosomen**

De fitness van een varken is gebaseerd op het gemiddelde van Seek, Cohesion en Alignment samen (fitness proportionate selection). Vervolgens de average fitness wordt dan berekend van alle varkentjes.

1. **Ouders selecteren**

Indien een fitness (truncation selection) van varken hoger is dan het gemiddelde, wordt deze gekozen als parent

1. **Nakomelingen genereren**

Uit de parents worden twee willekeurige parents gekozen die samen een child maken, waarbij een split point gekozen wordt van (1, 5) die deel van de parent chromosoom overneemt. Een parent kan meerdere childs hebben gemaakt door toeval.

Er is een mutatie kans van 1 op 100 per generatie die gemaakt wordt, waarbij een van de cohesion, seperation of alignment chromosoom een willekeurige waarde krijgt tussen 0.00 en 1.00

1. **Nieuwe populatie samenstellen**

De nieuwe populatie bestaat enkel uit nakomelingen, ouders worden dan uit de populatie gehaald. De opgeslagen chromosomen worden verwijdert.

# Q-learning

**Niet van toepassing.**