1 Inleiding

2 Beschrijving probleem en theorie

2.1 Omschrijving van het spel

Het is de bedoeling dat onze snake speelt tegen de snakes die onze tegenstanders hebben geprogrammeerd. Het veld bestaat uit hekjes, voedsel en uit andere spelers. De snake gaat dood als hij op andere spelers of hekjes terecht komt, en hij wordt n vakje langer als hij op voedsel terecht komt. Elke snake krijgt 100 punten voor het eten van voedsel, en 1000 punten als er een andere snake dood gaat terwijl hij zelf nog leeft. Ook krijgt de snake n punt per beurt dat hij een stap zet. Het doel is om het hoogste puntenaantal te halen. Belangrijk is op te merken dat als er een snake dood gaat, dat deze dan niet verdwijnt maar zijn lichaam blijft liggen. De dode slangen moeten dus ook in de gaten gehouden worden. De snakes bewegen in volgorde van puntenaantal; de speler met de minste punten beweegt als eerst. Als twee snakes dus in dezelfde beurt naar een vakje bewegen, dan gaat de snake met het meeste aantal punten dood. Dit zorgt ervoor dat de snakes ook niet in een gesloten cirkel achter zichzelf aan kan blijven lopen, er moet altijd een vakje tussen zijn hoofd en staart blijven.

- 1.2 objecten/andere spelers ontwijken

3 Strategie

3.1 A*

3.2 voedsel

Als het veld groot is, is veel voedsel eten een gunstige strategie. Als het veld erg klein is, beperkt dit de bewegingsvrijheid van de slang maar ook de bewegingsvrijheid van de tegenstanders. Als

3.3 Gangen

Het voorkomen dat er in een doolhof een doodlopende gang is, of dat er een gang is naar een gebiedje dat niet groot genoeg is om zelf helemaal in te kunnen. Het is belangrijk dat de snake hier niet in loopt, want dan loopt hij zichzelf klem. De snake moet dus zelf een gang kunnen detecteren. Een gang is een doorgang waar de slang n maal doorheen kan, als de slang door de gang loopt kan hij niet meer terug. De slang definieert een gang als twee hekjes die minimaal een afstand hebben van elkaar van 2. Ofwel, gangen zijn een variatie van de volgende vormen:

$$\begin{array}{ccc} . & \# & . \\ . & G & . \\ . & \# & . \end{array} \qquad \begin{array}{ccc} \# & G & . \\ . & G & \# \\ . & . & . \end{array} \tag{1}$$

Een gang is doodlopend als er maar n in- en uitgang is. Om dit te voorspellen wordt het veld opgedeeld in verschillende gebieden die worden gescheiden van elkaar door hekjes en gangen. Zo kunnen het aantal gangen die naar een gebied gaan worden geteld en aan de hand daarvan worden bepaald of een gebied 'veilig' is voor onze slang, zie 4.1.

- 1.3.3 hoe beslissen we de volgende stap

4 Gebruikte algoritmen en datastructuren

4.1 Floodfill

4.2 Dijkstra's algoritme

Om te bepalen wat de volgende stap van onze snake wordt, hebben we gebruik gemaakt van een variatie op Dijkstra's algoritme. Dit algoritme ....
De snake denkt vijf stappen vooruit. Elk vakje krijgt hierbij een bepaald aantal punten, en met behulp van een 'priority queue' wordt bepaald welke route het minste aantal punten heeft gekregen. Daarvan zet de snake de eerste stap. Dit gebeurt elke beurt opnieuw, omdat de positie van de (levende) snakes van andere spelers elke beurt verandert. Er zijn ... verschillende soorten vakjes die de snake tegen kan komen, namelijk:

- Een leeg vakje
- Voedsel
- Een gang, hierbij maken we onderscheid tussen een doodlopende gang en een doorgang tussen twee gebieden
- Een hekje
- Een andere snake

- 2.1 Informatie die we kunnen opslaan - 2.2 Hoe gebruiken we deze informatie - 2.3 Neural Network

3 Documentatie over het gebruik van de code 4 Eventuele resultate die behaald zijn met de code 5 Bronverwijzing naar gebruikte literatuur, websites etc.