

Maze Generator



Naam: Gino van de Graaf.

Version	Decription
0.1	Opzet en hoofdvraag en subvragen. Eerste uitwerking.

Inhoud

Voorwoord	3
Hoofdvraag.....	3
Sub vragen	3
Welke doolhof generators zijn er?.....	4
Heeft de criteria	4
Kruskal's Algorithm	4
Prim's Algorithm	5
Growing Tree algorithm	6
Haalden niet de criteria	7
Sidewinder algorithm.....	7
Aldous-Broder	7
Welke doolhof generator voelt het beste aan voor de electric maze?	8
Het maken van een prototype maze in code.....	9
Het testen/feedback van ander of de maze generator goed werkt.	10
Links	11

Voorwoord

Deze onderzoek gaat gepaard met een devlog om beter te kunnen demonstreren hoe bepaalde doolhoven generators werken.

Dit doolhof procedurele generator wordt gebruikt in de VR game Electric Maze. Maar het probleem is hoe ga ik een doolhof maken. Welken soorten generators zijn er? Welke is het beste om te gebruiken en kan snel een doolhof maken. Dit document zal al mijn antwoorden geven op deze vraag.

Hoofdvraag

Welke doolhof generator is het beste voor mijn elektrisch doolhof game?

Sub vragen

Welke doolhof generators zijn er? (Libary Literature study)

Welke doolhof generator voelt het beste aan voor de elektrisch doolhof? (Libary SWOT analysis)

Het maken van een prototype doolhof in code.

Het testen/feedback van ander of de doolhof generator goed werkt.

Verdere stappen.

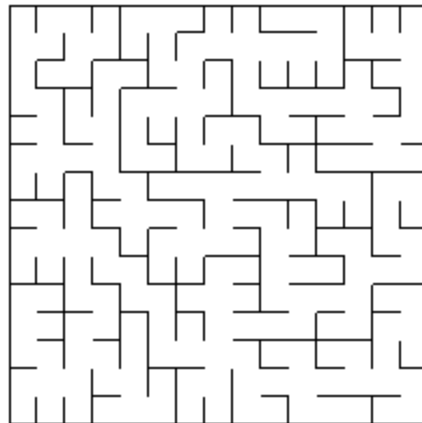
Welke doolhof generators zijn er?

Er zijn verschillende soorten doolhof generators die ik kan gebruiken en allenmaal hebben ze hun eigen manier van genereren. Dus heb wat regels voor mijzelf opgesteld waar het aan moet voldoen.

1. Een doolhof generator moet snel zijn met het genereren. En moet niet complex zijn in het programmeren.
2. Een doolhof generator moet niet voorspelbaar zijn.(dit kan bijvoorbeeld dat de uitgang van de route altijd hetzelfde is)
3. En het moet een beetje natuurlijk aanvoelen. We willen geen grote hobbels hebben waarbij je arm iedere keer moet draaien.
4. Er moeten zo weinig mogelijk dead ends creëren(dit betekent dat je de doolhof niet kan oplossen)

Heeft de criteria

Kruskal's Algorithm



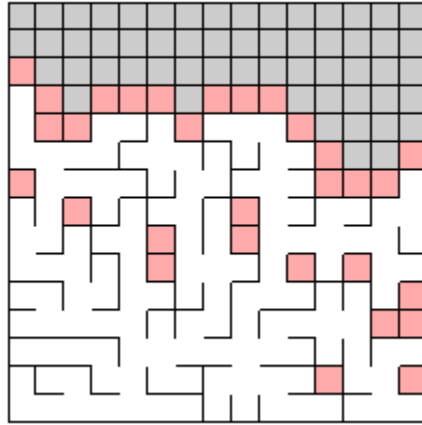
De kruskal algortime is een methode voor het produceren van een minimale spanning tree vanuit een gewogen graph. Het algoritme is eigenlijk een randomized algoritme.

- Doe alle randen in de grafiek in een grote verzameling. (Of, weet je, een set of iets dergelijks.)
- Haal de rand met het laagste gewicht eruit. Als de rand twee afzonderlijke bomen verbindt, verbind dan de bomen.
- Gooi anders die rand weg. Herhaal dit totdat er geen randen meer over zijn.

Dit algoritme is snel met generen van een doolhof. Het probleem van dit algoritme is dat veel korten eindpunten heeft.

Prim's Algorithm

Prim's benadert het probleem vanuit een andere hoek. In plaats van de hele graaf rand voor rand af te werken, begint het bij een punt en groeit het daarvandaan. De standaardversie van het algoritme werkt als volgt:

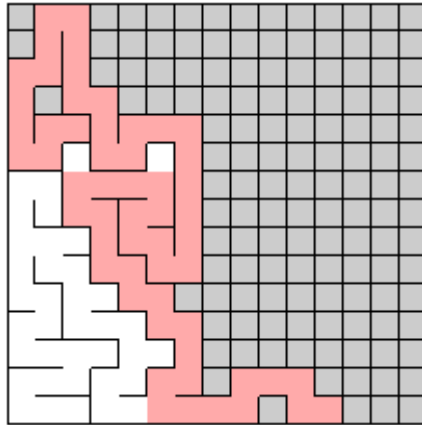


- Kies een willekeurige knoop uit de lijst en voeg deze toe aan een (initieel lege) verzameling.
- Kies de rand met het kleinste gewicht uit de lijst, die een knoop verbindt met een andere knoop die niet in zit.
- Voeg die rand toe aan de minimale spannende boom en voeg de andere knoop van die rand toe aan.
- Herhaal stappen 2 en 3 totdat V alle knopen van G bevat.

Het algoritme is best snel omdat hij meerder punten behandelt. Heeft wel hetzelfde probleem als de kruskal. En dat is dat ook korte eindpunten heeft.

Growing Tree algorithm

Het werkt bijna precies hetzelfde als het algoritme van Prim. Met nog een kleine wijziging kun je doolhoven genereren met eigenschappen van beide.



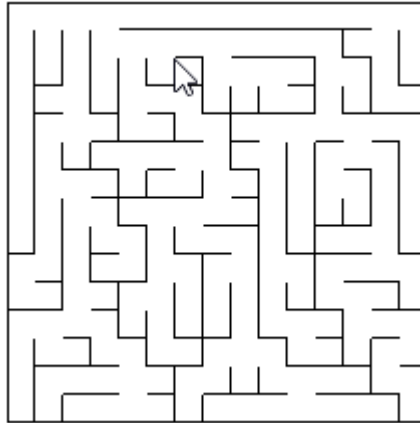
Zo werkt het:

- Laat C een lijst van cellen zijn, aanvankelijk leeg. Voeg één cel toe aan C, willekeurig gekozen.
- Kies een cel uit C en snij een doorgang naar een onbezochte buur van die cel en voeg die buur ook toe aan C. Als er geen onbezochte burens zijn, verwijder dan de cel uit C.
- Herhaal #2 totdat C leeg is.

Haalden niet de criteria

Dit zijn generators die niet haalden vanwege wat regels die ik had. Dat betekent niet dat het slechte algoritmes zijn. Maar ze voldoen niet aan de eisen wat ik wil.

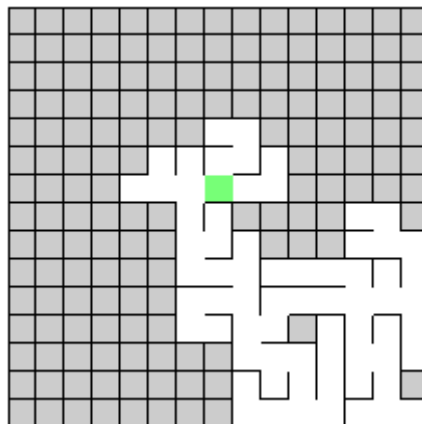
Sidewinder algorithm



het doolhof maakt het makkelijk om elke keer hetzelfde patroon te maken vooral boven in is een rechte weg die verbind met alle ander locaties. Het algoritme is wel snel met het generen van de maze.

Aldous-Broder

Leuk om naar te kijken maar het duurt zo lang dat het niet geschikt is. Het duurt zolang dat de prim doolhof 18 keer sneller is met generen.



Welke doolhof generator voelt het beste aan voor de electric maze?

Hierin doe ik een korte bench mark hoe snel ze zijn met generen van een doolhof. Alle doolhoven worden genereerd in een 16*16 grid. Dus 256 tegels.

Doolhof algoritme	Snelheid	Overall toegang	Makkelijk op te lossen	Maakt het doolhof natuurlijk gevoel
kruskal	6 seconden	ja	gemiddeld	Nee maakt korte eindpunten maar heeft wel langer opeenvolgende stukken
prim	5 sec	Ja	gemiddeld	
Growing tree	11 seconden	Ja	moeilijker	Meer bochten ook rechte stukken. Begin van generatie is random
sidewinder		Ja	Te makkelijk	Nee een gedeelte blijft hetzelfde met ieder generatie
Binary tree		Ja	Te makkelijk	Nee n gedeelte blijft hetzelfde met ieder generatie
Aldous-Broder	1 min en 28 second	ja	Moeilijker op te lossen	Ieder doolhof is uniek omdat het startpunt random is.

Het maken van een prototype maze in code.

Het testen/feedback van ander of de maze generator goed werkt.

[Links](#)