MMV TP3 - BTP - Construction de routes

Rédigé le 08/01/2021
par
Mathéo Dumont (matheo.dumont@etu.univ-lyon1.fr),
Jean-Baptiste Loutfalla (jean-baptiste.loutfalla@etu.univ-lyon1.fr)

***Abstract*—Utilisation de cartes de hauteurs pour implémenter la création de routes à l'aide d'un algorithme de plus court chemin.
Notre code source se trouve ici.**

I. Introduction

Nous utilisons Dijkstra, un algorithme de plus court chemin pour créer des routes tout en minimisant la pente des passages empruntés par celle-ci, ce qui revient à faire des route moins pentues. Notre travail s'inspire de celui fait par notre professeur ainsi que de ses collègues [Gal+10].

II. Travail réalisé

A. La base

Notre carte de hauteur / terrain est représentée avec une grille, contenant la hauteur pour chaque point (i, j) du terrain. Pour obtenir une route entre un point de départ et un point d'arrivée, nous utilisons un algorithme de plus court chemin, pour lequel nous devons créer un graphe.

B. Placement des extrémités de la route

Le point de départ et le point d'arrivée sont placés en utilisant la vue 2D de la carte de hauteur. Pour déterminer l'emplacement de ces points sur notre terrain, nous avons converti les coordonnées de la souris relative à l'élément affichant notre image lors du clic aux coordonnées du pixel correspondant dans notre image, ce qui nous permet après de déterminer la position sur notre terrain pour réaliser nos calculs.

C. Création du graphe d'adjacences

Le terrain est représenté comme un graphe où chaque cellule est un noeud et le passage d'une cellule à une autre est une arête qui relie ces deux noeuds. On paramétrise le graphe ainsi :

1) M définit la taille d'un masque $M \times M$, centré sur la cellule concernée qui représente tous les voisins accessible depuis un noeud, un M_0 permet d'atteindre tous les voisins à une cellule de distance ($distance = 1$) sur le terrain.
2) $slope\_coeff$ définit l'importance qu'aura la pente lors du calcul du chemin. Un $slope\_coeff$ égal à 0 revient à chercher le plus court chemin sans prendre en compte la pente et une valeur de 1 revient à considérer la pente sans l'accentuer.

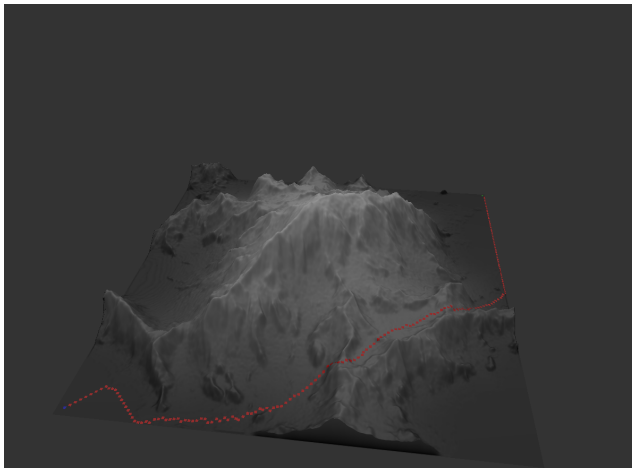

Fig. 1: Route construite avec un paramètre $slope\_coeff$ fort (carte Spike).

D. Implémentation

C'est un ajout fait dans notre implémentation de base des cartes de hauteurs. Nous construisons le graphe d'adjacences comme cité plus haut, puis, avec un Dijkstra pris sur Rosetta Code par soucis de simplicité, nous calculons le plus court chemin. Nous mettons à jour l'affichage en sélectionnant dans notre mesh les couleurs des sommets à mettre à jour. La création du graphe est de complexité $\mathcal{O}(n \times M)$, avec $n = height \times width$, la taille de la grille. La complexité de Dijkstra est, comme indiqué sur le site $\mathcal{O}(n \times logV)$ où V est le nombre d'arêtes, soit $n \times M$.

Pour donner un ordre de grandeur, sur une grille 1081×1081 pixels, avec $M = 4$, la création du graphe, le calcul de Dijkstra et la mise à jour de l'affichage prennent ensemble 5sec environ.

E. Discussion

En utilisant un $slope\_coeff$ élevé, le plus court chemin sera fait en prenant soin d'éviter tout escarpement et pente (Fig. 1). Alors qu'avec un faible $slope\_coeff$, notre algorithme emploiera plus facilement la technique du *"je vais tout droit"*, en empruntant des pentes escarpées (Fig. 2).

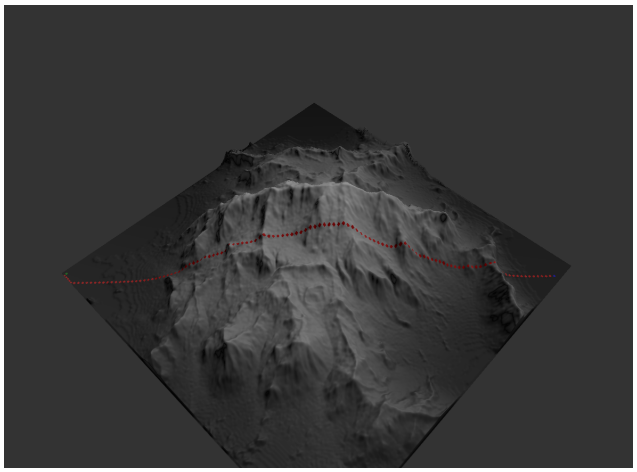

Fig. 2: Route construite avec un paramètre $slope\_coeff$ faible (carte Spike).

Un autre exemple sur une carte avec beaucoup de relief et une accentuation de $slope\_coeff$ (Fig. 3).

F. Améliorations

Pour commencer, notre masque de taille $M \times M$ permet de *"sauter"* certains pics. Comme on peut le voir sur cette figure (Fig. 2), vers la fin de la route, le chemin traverse/saute le pic, alors que la pente est élevée et qu'en théorie, il devrait l'éviter. Cela s'explique par le fait qu'étant donné notre grand masque ($M = 4$), il *"saute"* sur un point plus bas, car celui-ci a une pente plus faible et que le coût réel où l'on fait tout le chemin est ignoré. Une amélioration du masque serait de ne pas répéter plusieurs arêtes allant dans la même direction mais plus loin. On garderait ainsi un nombre de direction intéressant.

Une autre amélioration possible serait de conserver le graphe d'adjacences réalisé lors du calcul d'une première route sur notre carte de hauteur. En effet, pour le moment nous le recalculons intégralement à chaque fois que l'on veut créer une route ce qui peut être lourd quand on travaille uniquement sur une seule carte de hauteur.

III. Reproduire nos résultats

Pour nos tests, nous avons utilisé différentes cartes de hauteur de différents endroits :

1) Spike, un lieu anonyme ayant comme superficie 15km² et une hauteur variant entre 0m et 5000m
2) Vanoise, une partie du parc de Vanoise ayant comme superficie 18km² et une hauteur variant entre 297m et 3825m

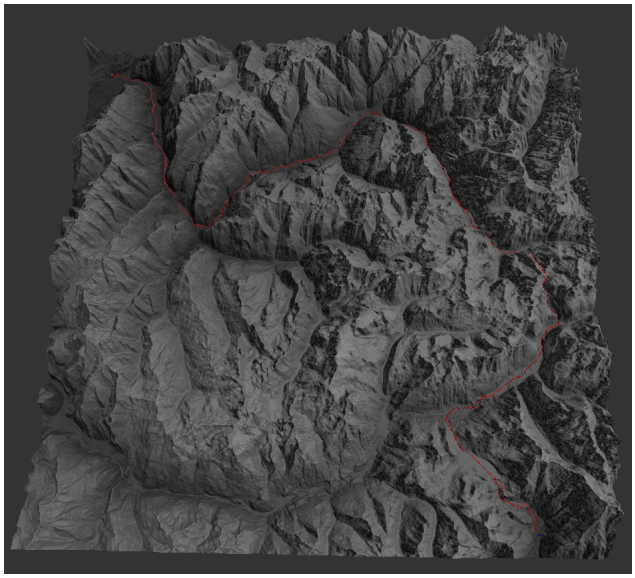

Fig. 3: Exemple de route sur le terrain Vanoise.

Références

[Gal+10] E. Galin et al. "Procedural Generation of Roads". In: *Computer Graphics Forum* 29.2 (2010), pp. 429–438. DOI: https://doi.org/10.1111/j.1467-8659.2009.01612.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2009.01612.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2009.01612.x.