18298 KARDID Anwar

Génération procédurale des terrains virtuels dans les jeux vidéos

La génération procédurale permet de créer des mondes dynamiques avec une diversité infinie. Elle optimise les ressources, diminue les coûts de développement et procure des expériences uniques. Les jeux comme Minecraft[6] et NoMan's Sky[8] illustrent la puissance de cette technique.

La génération procédurale, largement employée dans les jeux vidéo, offre la création de terrains aux textures et reliefs complexes. Idéale pour les jeux sportifs, elle permet la conception de terrains variés et réalistes tels que stades de foot, forêts diversifiées et circuits de courses...

Positionnement thématique (ÉTAPE 1) :

- *INFORMATIQUE (Informatique pratique)*
- *INFORMATIQUE (Technologies informatiques)*
- *MATHEMATIQUES (Géométrie)*

Mots-clés (ÉTAPE 1) :

Mots-clés (en français)	**Mots-clés (en anglais)**
Bruit de Perlin	*Perlin Noise*
Déplacement du point médian	*Midpoint displacement*
Génération procédurale	*Procedural content generation PCG*
Graphisme informatique	*Computer graphics*
Géométrie algorithmique	*Algorithmic geometry*

Bibliographie commentée

• Dans l'industrie des jeux vidéo, la génération procédurale des terrains demeure importante. Les jeux sportifs en particulier nécessitent des environnements diversifiés et réalistes dont la création manuelle est coûteuse et prend beaucoup de temps. Les algorithmes procéduraux offrent une solution efficace en réduisant le temps de conception et en fournissant des résultats visuels intéressantes, tout en optimisant les ressources.

• Concernant l'algorithme de Perlin Noise, Kenneth Perlin l'a initialement développé en 1983 dans un contexte totalement différent : la création d'effets visuels pour le film "TRON" [7] Son

objectif initial était d'apporter des éléments graphiques à ce film. Par la suite, Perlin a transformé cet algorithme en une version capable de générer des valeurs pseudo-aléatoires, connue sous le nom de Perlin Noise, comme décrit dans son article scientifique intitulé "An image synthesizer. ACM SIGGRAPH" [1]. Finalement, une version plus aboutie et performante a été mise en œuvre, connue sous le nom de Simplex Noise[9].

• Cet algorithme est largement utilisé dans des jeux très populaires tels que Minecraft [6] et No Man's Sky [8] , qui ont généré des revenus dépassant les 2.8 billions de dollars. Cette évolution soulève plusieurs questions intéressantes : Quels critères techniques évaluent les performances des algorithmes aujourd'hui ? Jusqu'où peut-on atteindre en termes de réalisme virtuel des jeux vidéo avec nos connaissances scientifiques actuelles ? Comment intégrer efficacement cet algorithme dans les jeux vidéo en optimisant les performances et en assurant une meilleure interaction avec les joueurs ?

• Nous baserons notre approche sur l'article [2] rédigé par Adrian Biagioli, un développeur certifié chez Unity Technologies, ainsi que sur les informations fournies sur le site web [3], pour explorer les principes de cet algorithme.

• Dans une autre part, Alain Fournier[10], né à Lyon, France, a joué un rôle majeur dans le développement de l'algorithme du Midpoint Displacement. L'algorithme trouve ses racines dans le domaine de la modélisation graphique et de la simulation, et son histoire remonte aux années 1980. L'idée générale derrière le Midpoint Displacement repose sur une génération récursive des terrains. Nous l'aborderons du point de vue de l'article [5] publié en May, 2003par Min-Fang Grace Tsai une étudiante chercheure à University of Toronto, Department of Mathematics.

Problématique retenue

- Comment aborder les défis de la conception numérique des terrains virtuels dans les jeux vidéo?

- Et à quel point on peut combiner entre performance, qualité visuelle et optimisation des ressources informatiques?

Objectifs du TIPE du candidat

1 - D'abord, générer des cartes de hauteurs par deux approches différentes :

a) Bruit de perlin, b) MidPoint displacement

2 - Ensuite, évaluer les qualités et exposer les défauts de chaque approche.

3 - Finalement, confronter ces deux algorithmes dans une simulation locale.

Références bibliographiques (ÉTAPE 1)

[1] PERLIN, K. : An image synthesizer *: ACM SIGGRAPH Computer Graphics, 19(3), 287-296.*

[2] ADRIAN BIAGIOLI : Analyse – géométrie –application de perlin *: https://adrianb.io/2014/08/09/perlinnoise.html*

[3] SCRATCHPIXEL, : Analyse – géométrie –application de perlin *: https://www.scratchapixel.com/lessons/procedural-generation-virtual-worlds/perlin-noise-part-2/perlin-noise.html*

[4] THE CRAFT OF CODING : Implementation du Perlin, *: https://craftofcoding.wordpress.com/2021/07/09/midpoint-displacement-in-2d/*

[5] MIN-FANG GRACE TSAI : MAT 335 Project Fractal Landscapes *: https://www.sfu.ca/~rpyke/335/projects/tsai/report1.htm*

[6] ALAN ZUCCONI : Explication de fonctionnement du jeu Minecraft *: https://www.alanzucconi.com/2022/06/05/minecraft-world-generation/*

[7] IMDB : Cinématographie *: https://www.imdb.com/title/tt0084827/?ref_=ext_shr_lnk*

[8] NO MAN'S SKY : SiteWeb official du jeu NoMan's Sky *: https://www.nomanssky.com/*

[9] STEFAN GUSTAVSON : Simplex noise demystified *: https://www.researchgate.net/publication/216813608_Simplex_noise_demystified*

[10] ALAIN FOURNIER, DONALD S. FUSSELL, AND LOREN C. CARPENTER : Computer render of stochastic models *: Commun. ACM, 25(6):371-384, 1982*