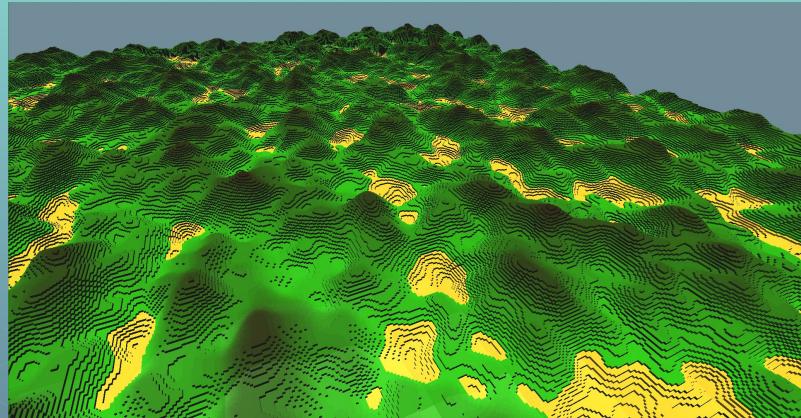




Moteurs de jeux

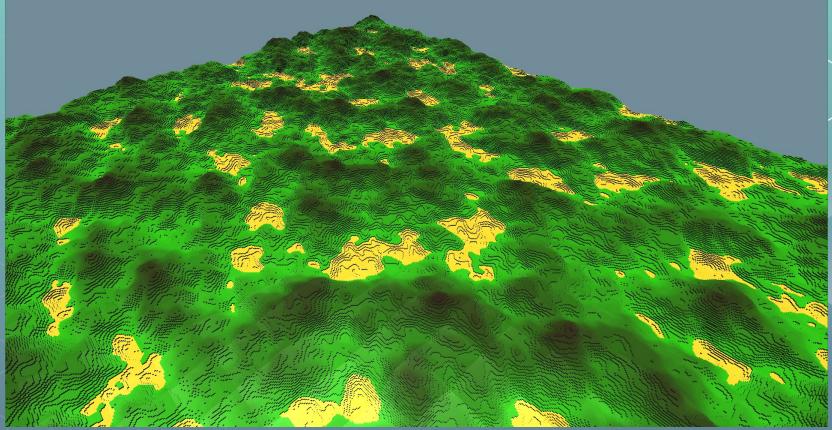


Wav3 Engine

Créé et présenté par

ISNEL MAXIME et ODORICO THIBAULT

Enjeu du projet



- **Faire un moteur de jeu adaptatif**
- **Faire un moteur à base de voxel**
- **Génération de terrain procédural**
 - **terrain infini**
- **Optimisation pour être temps réel**
- **Avoir un gameplay qui permet d'explorer**

Outils utilisés

- **Langage c++ (version >= c++11)**
- **OpenGL avec glad**
- **GLM**
- **GLFW3**
- **ImGui**
- **Cmake**
- **Git (dépot sur Github)**



Cross compile

Travis-ci.com



✓ master Merge branch 'develop'

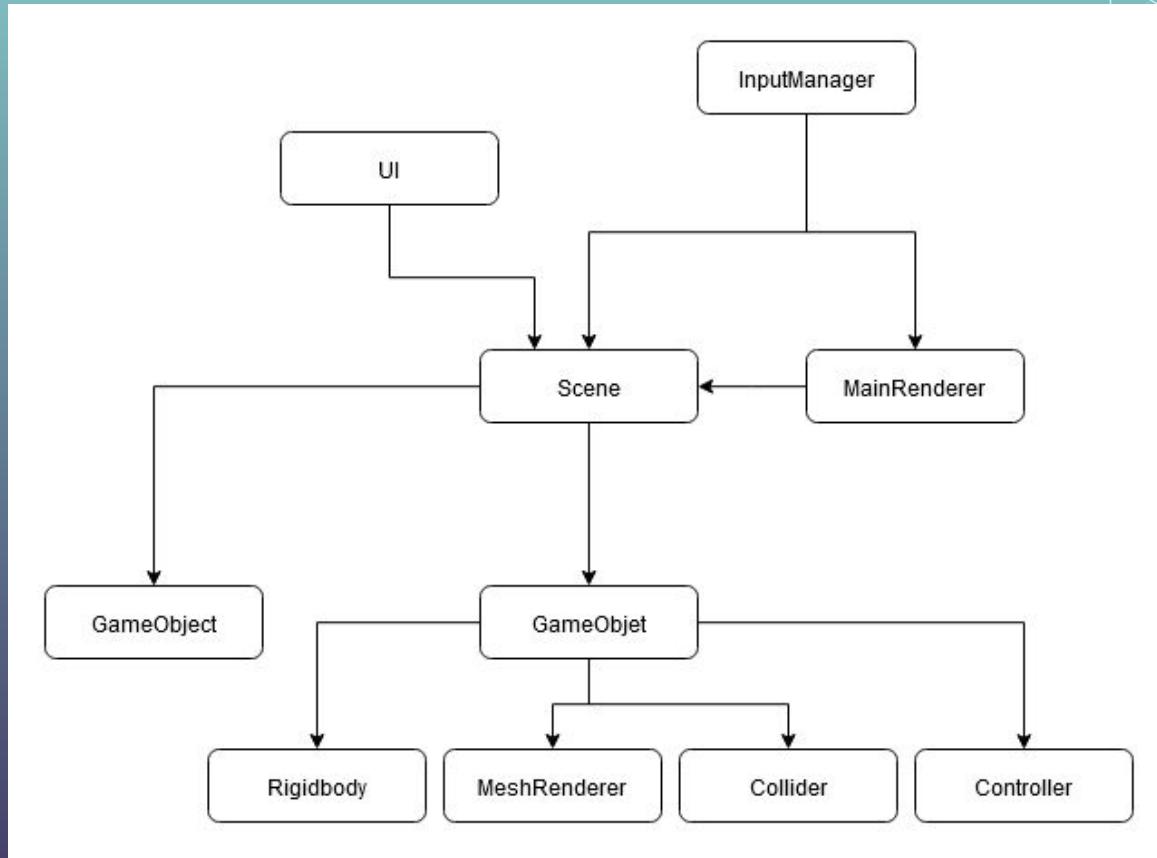
Commit 24d1e6a · Compare 7a1589c..24d1e6a · Branch master · maximel

113 passed · Ran for 5 min 18 sec · Total time 10 min 44 sec · a day ago · Restart build

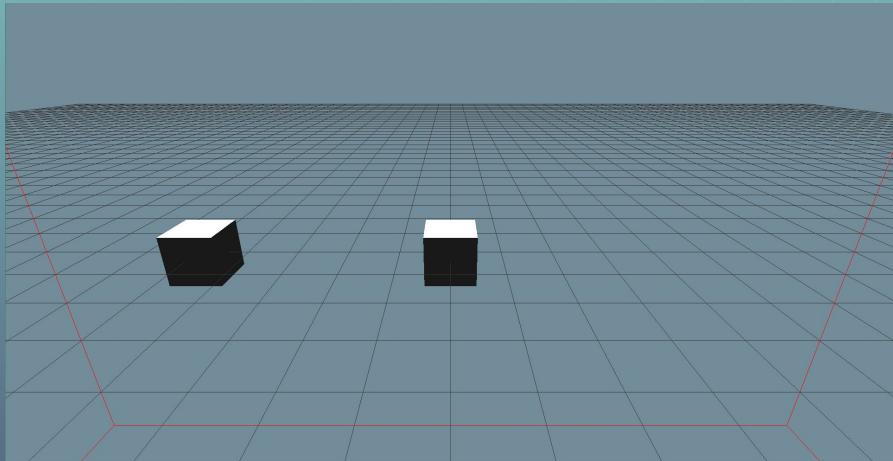
Build jobs · View config

Build	Time
✓ # 113.1 AMD64 Linux Build Debug	5 min 18 sec
✓ # 113.2 AMD64 MacOS Build Debug	1 min 17 sec
✓ # 113.3 AMD64 Windows MVS 2017 Build Debug	1 min 9 sec
✓ # 113.4 AMD64 Windows MinGW Build Debug	1 min 30 sec
✓ # 113.5 AMD64 Linux Build Release	1 min 18 sec
✓ # 113.6 AMD64 MacOS Build Release	1 min 29 sec
✓ # 113.7 AMD64 Windows MVS 2017 Build Release	1 min 14 sec
✓ # 113.8 AMD64 Windows MinGW Build Release	1 min 21 sec

Structure du moteur



Mode éditeur et mode jeu

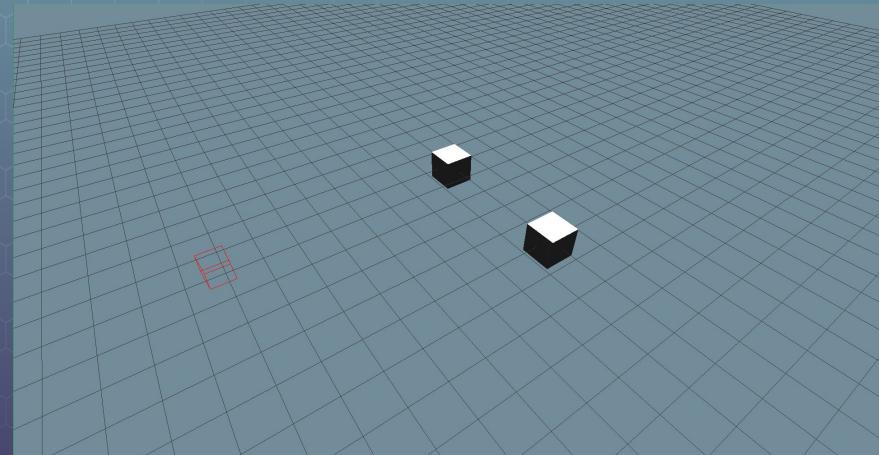


Vue du jeu

Démonstration



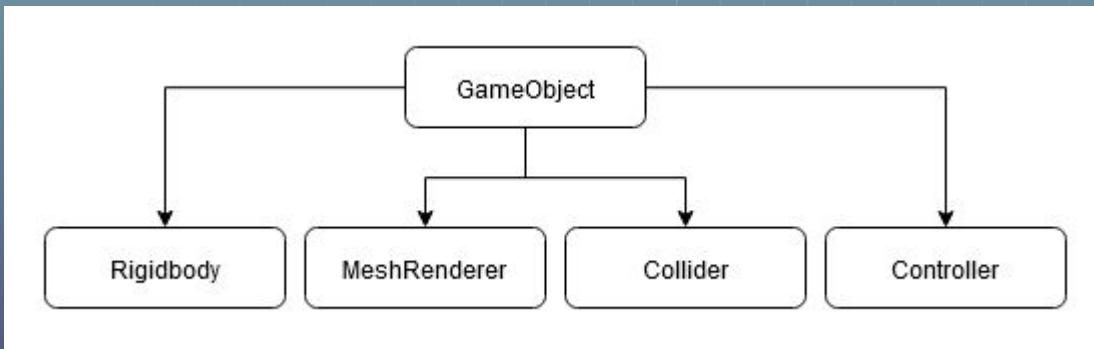
Vue du éditeur



Structure à base de composants

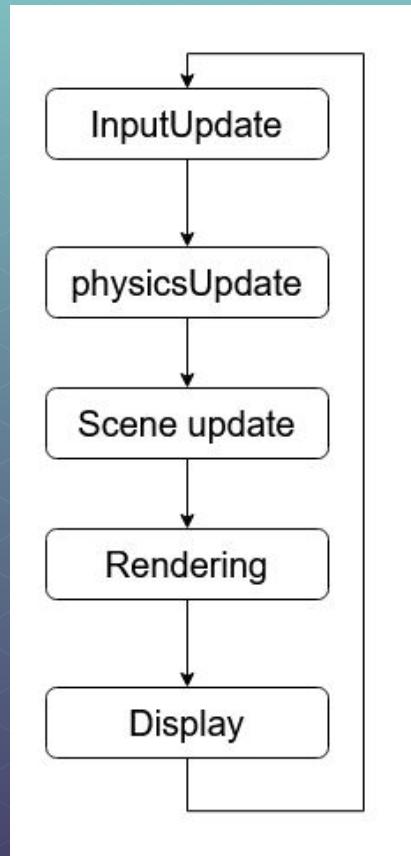


- Séparer le code en fonctionnalités
- Éviter l'héritage infini
- Changer le comportement au runtime



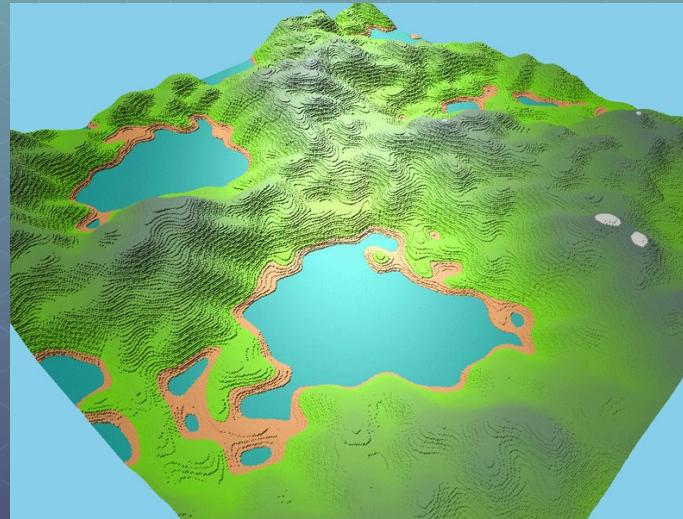
Démonstration

Game Loop



Voxel engine

- Monde à base de cube
- Génération procedurale
 - monde infini



Voxel engine

Problématiques

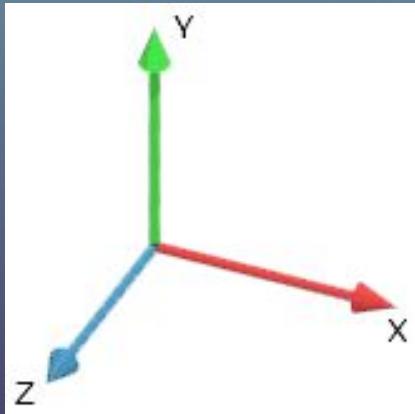
- Gestion de beaucoup de voxels
- Affichage de beaucoup de voxels
- Chargement d'un terrain infini
- Modification en temps réel



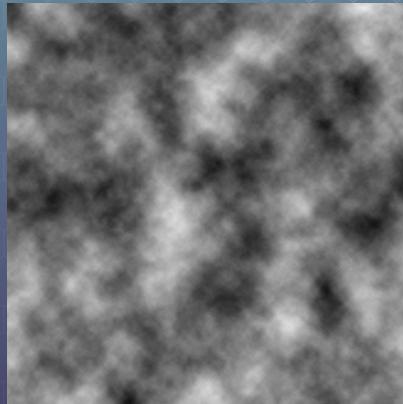
Optimisations

Division en chunk

- Position
- Bruit de perlin
- Tableau Cubique



+



-

A 3D perspective view of a large 3D grid of numbers, representing a cube of data points. The grid is labeled with indices from 0 to 9 along each axis.

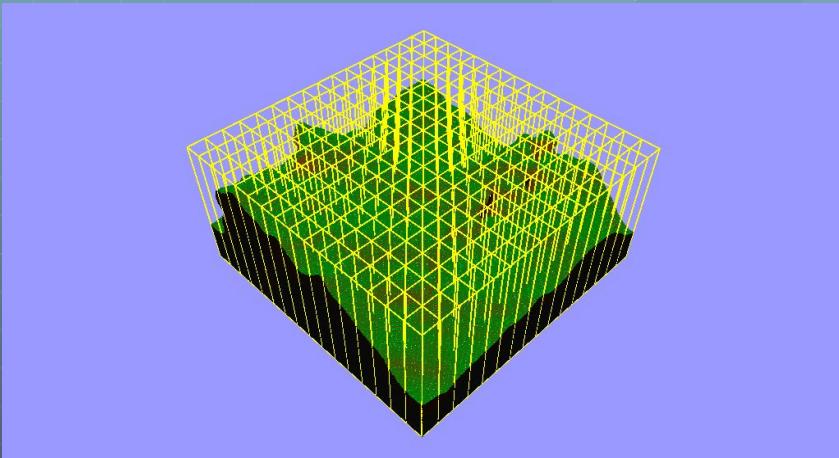
Index	0	1	2	3	4	5	6	7	8	9
0	65,340	12,483	138,189	902,960	633,877	103,902	236,781	601,691	329,274	913,534
1	5,246	424,642	650,380	821,254	866,122	105,802	4,567	236,781	733,611	329,274
2	89,678	236,781	601,691	329,274	913,534	658,305	128,782	705,588	978,155	520,702
3	103,902	4,567	733,611	263,010	85,550	2,778	53,028	523,784	556,801	847,107
4	2,778	658,305	128,788	978,155	620,702	45,024	55,058	775,046	548,322	85,587
5	45,024	55,058	705,586	89,672	384,605	780	67,538	834,553	638,108	56,083
6	780	47,538	523,784	556,801	617,107	32,667	550,850	775,046	548,322	41,123
7	32,667	350,890	834,753	638,108	85,188	145,582	557,739	515,048	418,482	56,083
8	56,083	145,582	775,040	548,322	756,587	41,123	443,542	537,738	513,048	418,482
9	41,123	543,542	537,738	513,048	418,482					



Optimisations

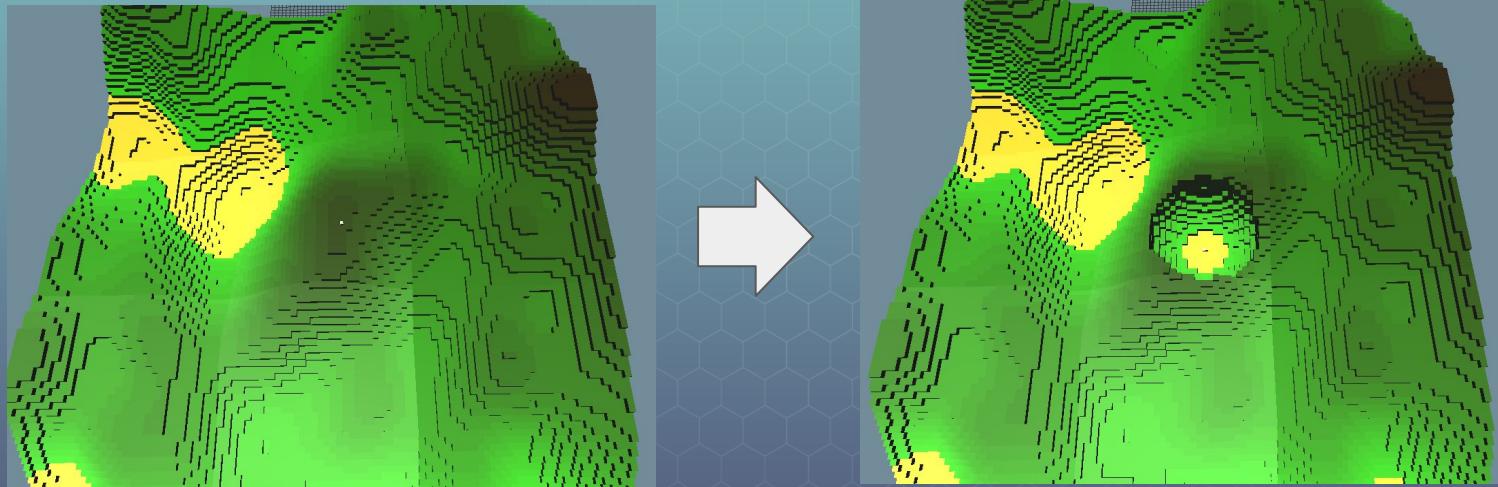
Division en chunk

Index	0	1	2	3	4
0	65,340	12,483	138,189	902,960	633,877
1	5,246	424,642	650,380	821,254	866,122
2	89,678	236,781	601,691	329,274	913,534
3	103,902	4,567	733,611	263,010	85,550
4	2,778	658,305	128,788	978,155	620,702
5	45,024	55,058	705,586	89,672	384,605
6	780	47,538	523,784	556,801	617,107
7	32,667	350,890	834,753	638,108	85,188
8	56,083	145,582	775,040	548,322	756,587
9	41,123	543,542	537,738	513,048	418,482



Optimisations

Division en chunk

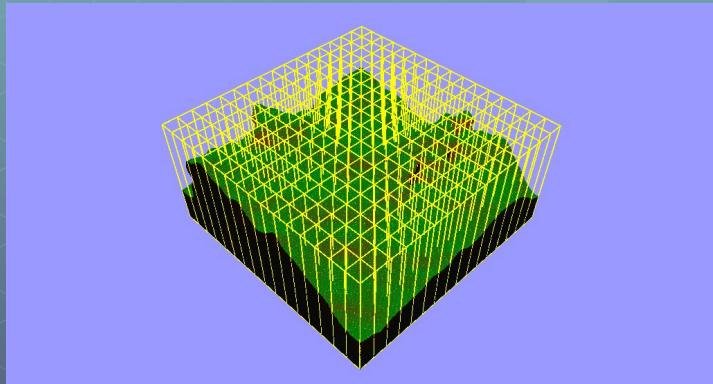


- **Modification locales moins coûteuses**
- **Maillage régénéré à la volé quand nécessaire**

Optimisations

Génération de la géométrie

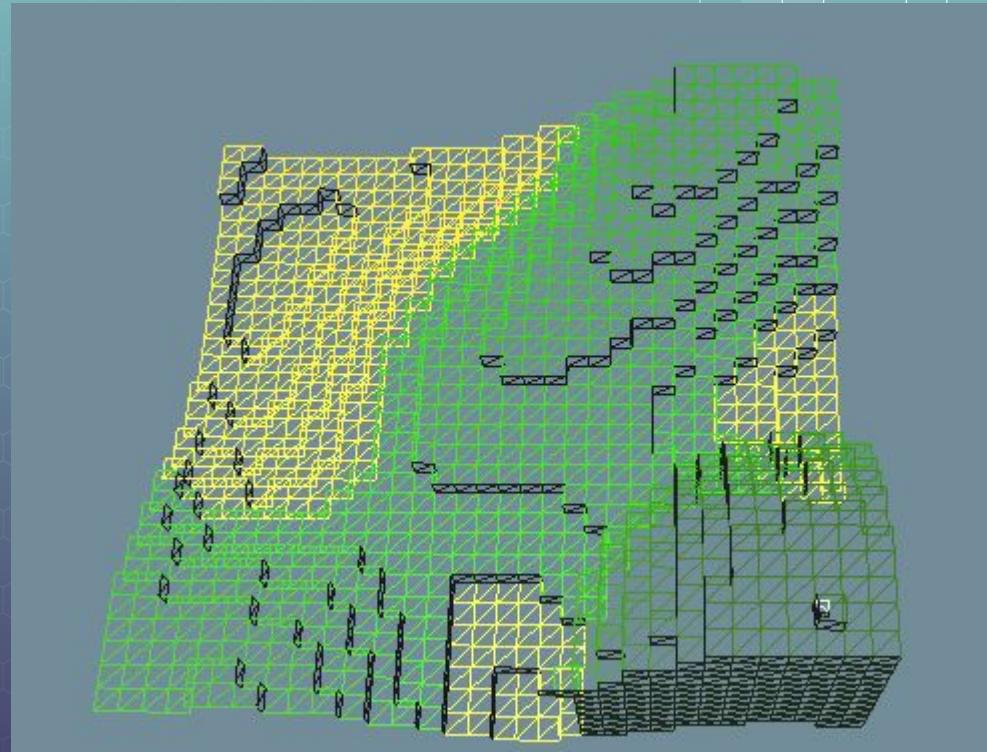
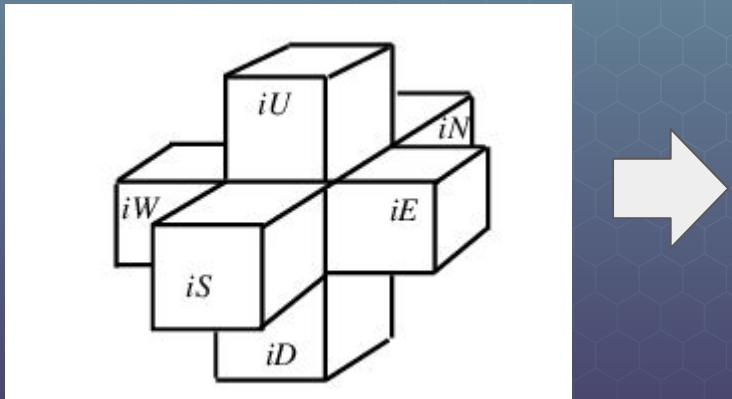
- **Cube = 6 faces = 12 Triangles**
- **Chunk $32 \times 32 \times 32 = 393216$ Triangles**
- **Communication GPU Couteuse**



Optimisations

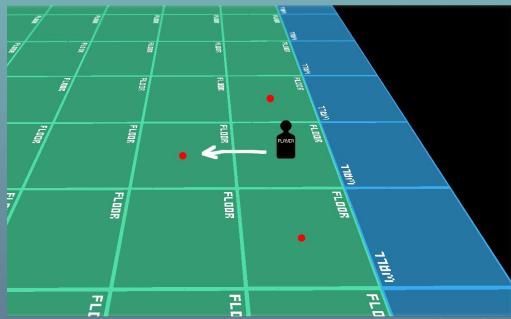
Génération de la géométrie

- Ignorer les voxels vides
- Ignorer les faces partagées

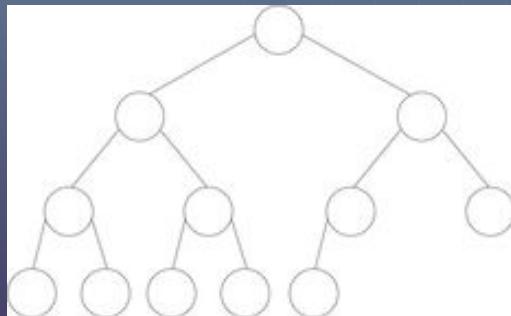


Optimisations

Chargement des chunks



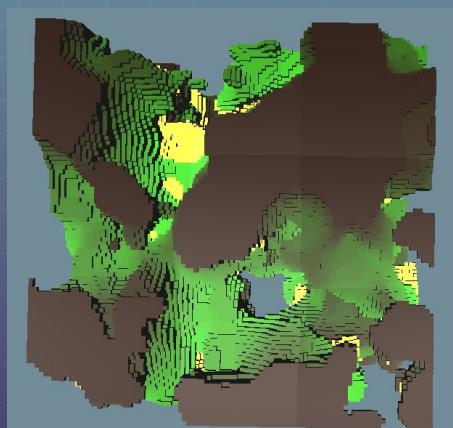
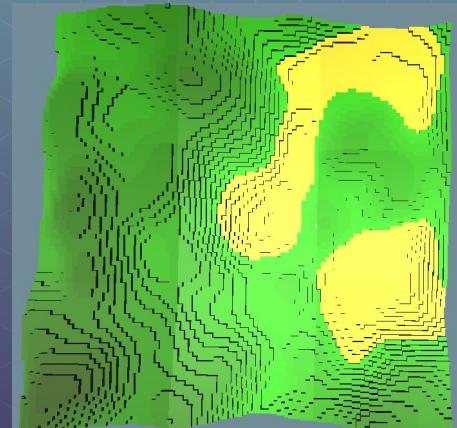
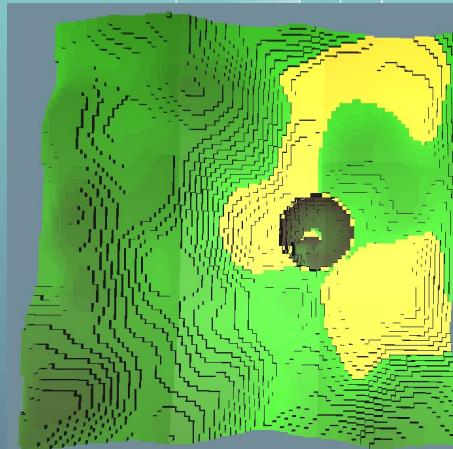
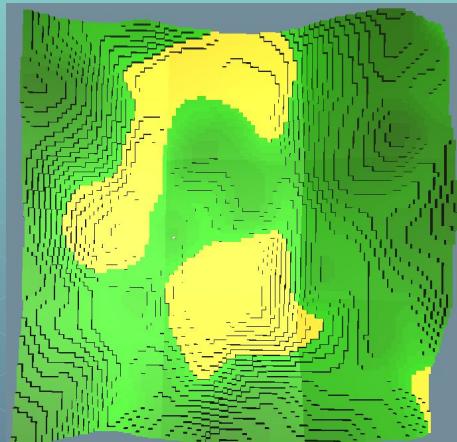
+



Optimisations

Chargement des chunks

- Déplacement possible sur tous les axes
- Génération uniquement des chunks



Gameplay

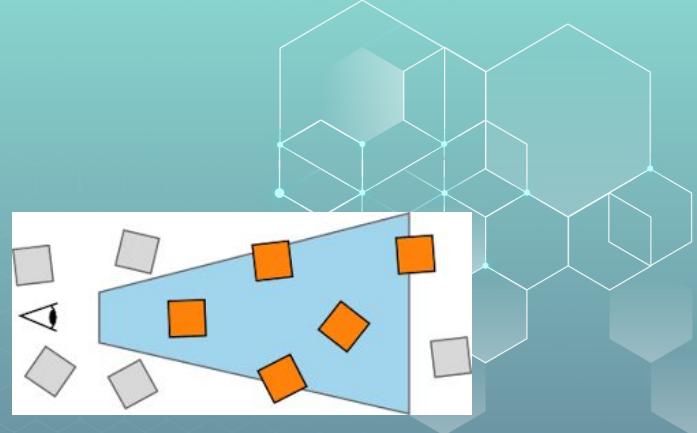
- Mouvement de camera
- Mouvement du personnage
- Explosions

Démonstration



Améliorations

- génération procédurale plus riche
- implémentation “frustum culling”
- collisions entre objets
- meilleur gestion de la caméra
- ajout d'ennemis se baladant dans le monde
- création d'un jeu type tower defense





MERCI POUR VOTRE ATTENTION