Labyrinthe & Génération Procédurale

LD M2

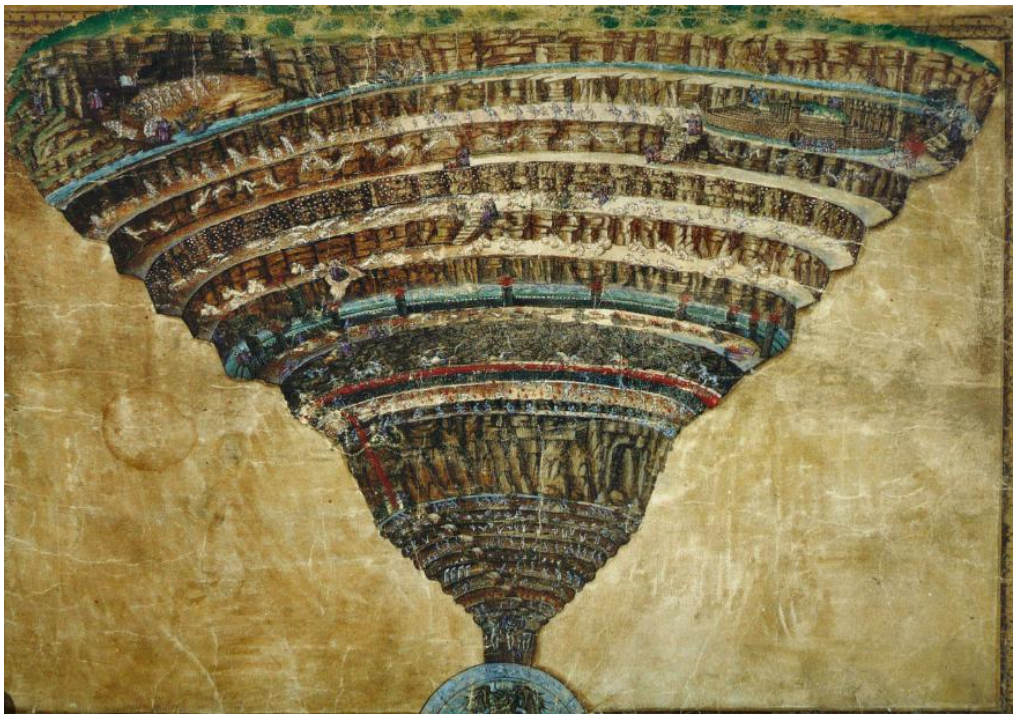
Ralph KMEID

Barthélémy JOLY

Sommaire

I - Intentions Narratives

Le joueur incarne un prisonnier considéré hérétique par un culte religieux. Au début, une cinématique montre les religieux ouvrant sa cellule. On croit être libre, mais il s'agit d'un piège : la prison est en réalité une structure infinie et cyclique. On y croise d'autres prisonniers condamnés à errer pour l'éternité.



Nous nous appuyons également sur **l'Inferno** de Dante Alighieri dans *La Divine Comédie*. Dans cette référence littéraire, le pécheur descend cercle après cercle, et chaque niveau est plus terrible que le précédent : plus le péché est grave, plus la souffrance est raffinée, cruelle et explicitement adaptée au crime commis. Cette structure en spirale descendante crée un effet d'enfermement et d'oppression croissante.

Nous reprenons ce principe dans notre représentation du labyrinthe-prison : le labyrinthe devient une version profane de l'Enfer, où l'âme se perd avant le corps. Cet imaginaire s'accorde avec l'effet "noir" que nous voulions produire : une atmosphère sombre, fataliste. Cependant, la progression de notre labyrinthe est inversée : on commence tout au fond et il faut remonter pour atteindre la lumière.

La prison est donc le labyrinthe lui-même : un espace construit pour tourmenter les prisonniers, condamnés à chercher éternellement une sortie. C'est une **torture psychologique** qui conduit à la folie.

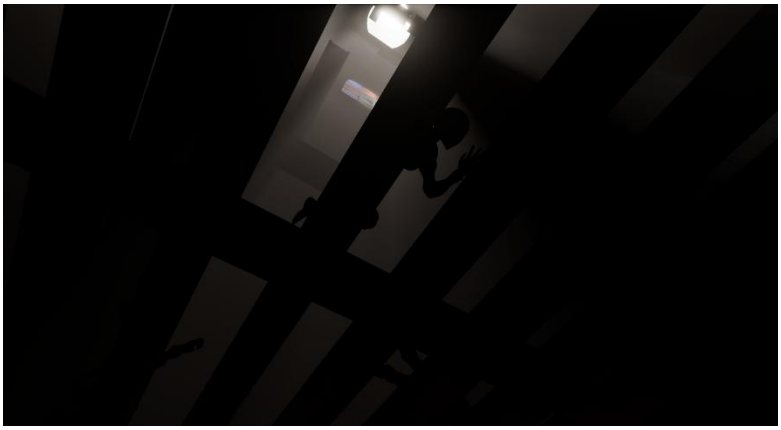
Mais le labyrinthe n'a pas pour seule fonction de punir. Il sert aussi un objectif implicite : la **conversion**. En égarant les hérétiques, il ambitionne de les ramener au « bon chemin » et de les faire revenir à la doctrine officielle.

Finalement, le labyrinthe représente une **critique des fondamentalismes religieux et des cultes fanatiques**. La religion pratiquée par ce culte n'est pas mauvaise à l'origine : son message était sain, humaniste, porteur de paix. Mais il a été corrompu, déformé par l'organisation qui s'en est emparée, modelé pour servir des intérêts terrestres, politiques, sociaux ou économiques. Ce détournement du sacré a transformé la foi en instrument de contrôle et de violence.
Ainsi, le joueur est considéré comme hérétique non pas parce qu'il a péché, mais parce qu'il refuse une **vérité falsifiée**.

II – Narration Environnemental

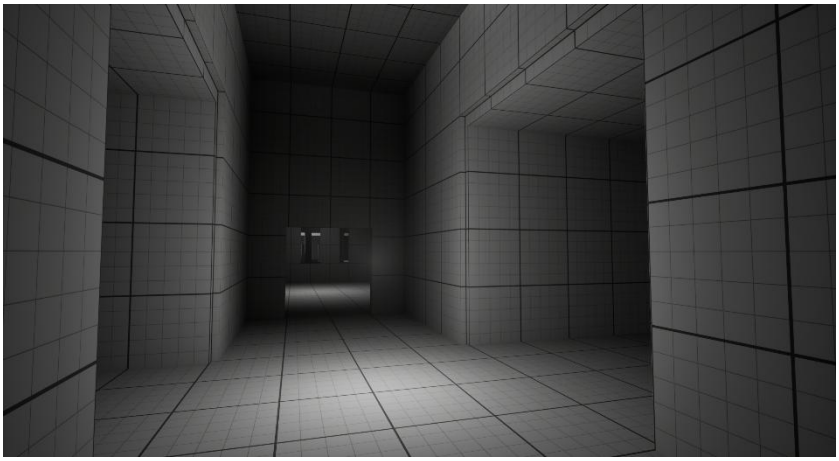
Le labyrinthe repose sur une **progression narrative et architecturale** qui raconte une histoire uniquement par l'espace.

Minuscule et écrasé, avec un défi :

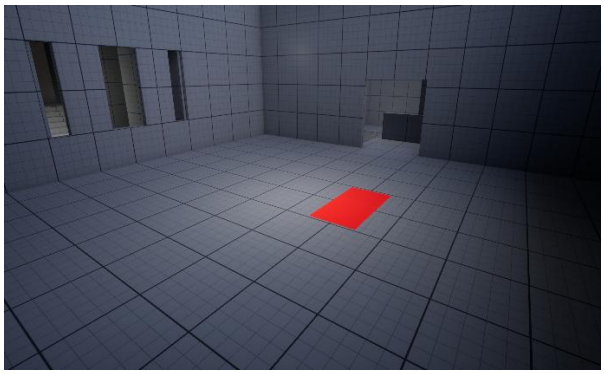


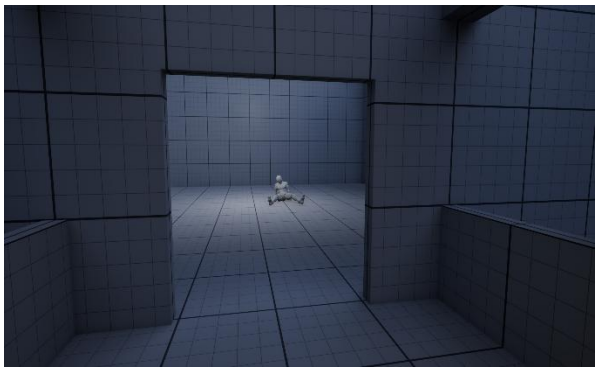
Tout commence avec le premier choc visuel : au-dessus de nous, une grille d'où pendent des corps sans vie, et au-dessous, une autre grille qui donne sur une salle de torture. Dès l'entrée, le joueur est pris en tenaille entre la mort passée et la souffrance à venir. On aperçoit le puits central depuis le bas : immense, vertical, écrasant. On se sent **minuscule**, enfermé, comme condamné à l'éternité du lieu. Pourtant, tout en haut, la lumière traverse l'obscurité. Peut-être une sortie. Peut-être un espoir. Et l'on voit les ponts suspendus, les plateformes, le labyrinthe à gravir : le défi est posé.

Torture, péché et punition :

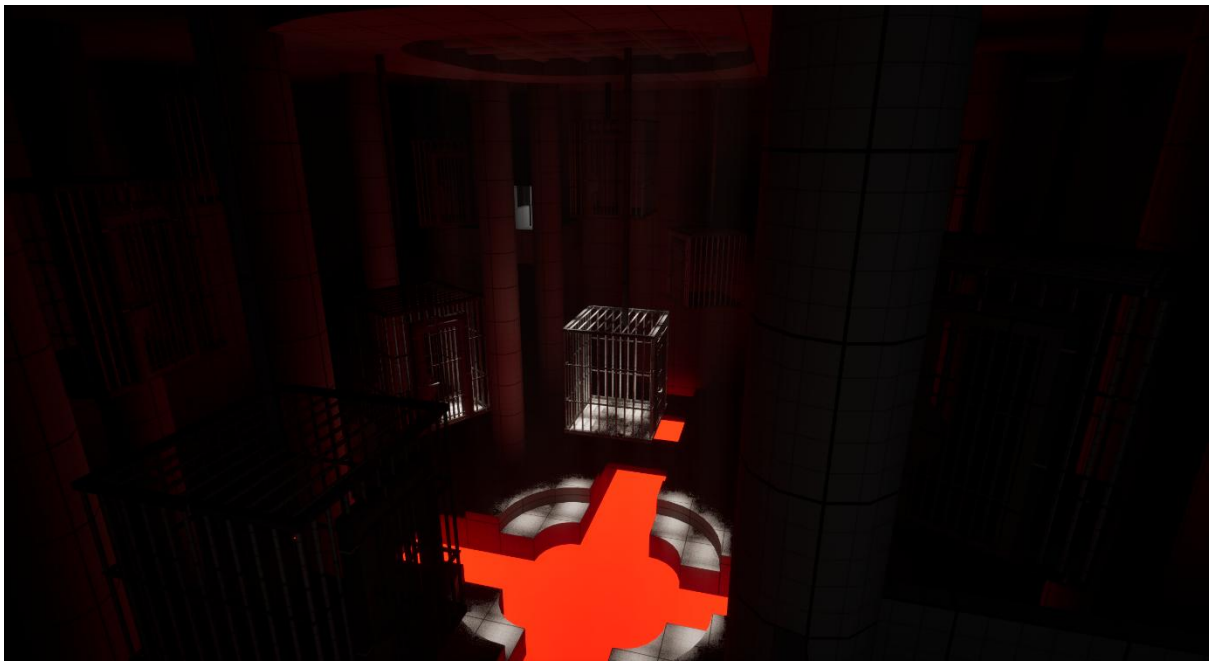


Les premiers couloirs sont étroits et sombres. Plusieurs entrées, plusieurs sorties, mais toutes se ressemblent. On trouve du décor éparpillé, comme cages, chaînes, sang, d'instruments de torture et corps abandonnés. L'atmosphère est oppressante, métallique, suffocante. Cet espace rappelle au prisonnier la faute, le péché et la punition.





On tombe sur la première salle principale : la salle de torture.



Ici, la violence n'est plus suggérée, elle est affichée et explicite. Du sang circule à travers des conduits ouverts, les murs portent les marques des coups, des cages oscillent au-dessus du vide, et des corps mutilés tapissent le sol. Des instruments de torture reposent encore sur des tables métalliques, certains souillés, d'autres prêts à servir.
Tout indique clairement la fonction du lieu : punir, briser, réduire au silence. Le joueur comprend qu'il se trouve au niveau le plus bas du labyrinthe, dans un enfer bien réel.

Église illusoire :

Puis, soudain, un **contraste brutal** : l'espace s'ouvre sur une salle inspirée des cathédrales européennes.

Des vitraux illuminent vers le puit central, des encensoirs brûlent, des cendres et des reliques reposent au sol. La lumière filtre, douce, presque sacrée. C'est un moment de respiration, un changement de rythme, un piège narratif.
Tu as vu la torture provoquée par tes péchés… mais regarde ce que tu pourrais

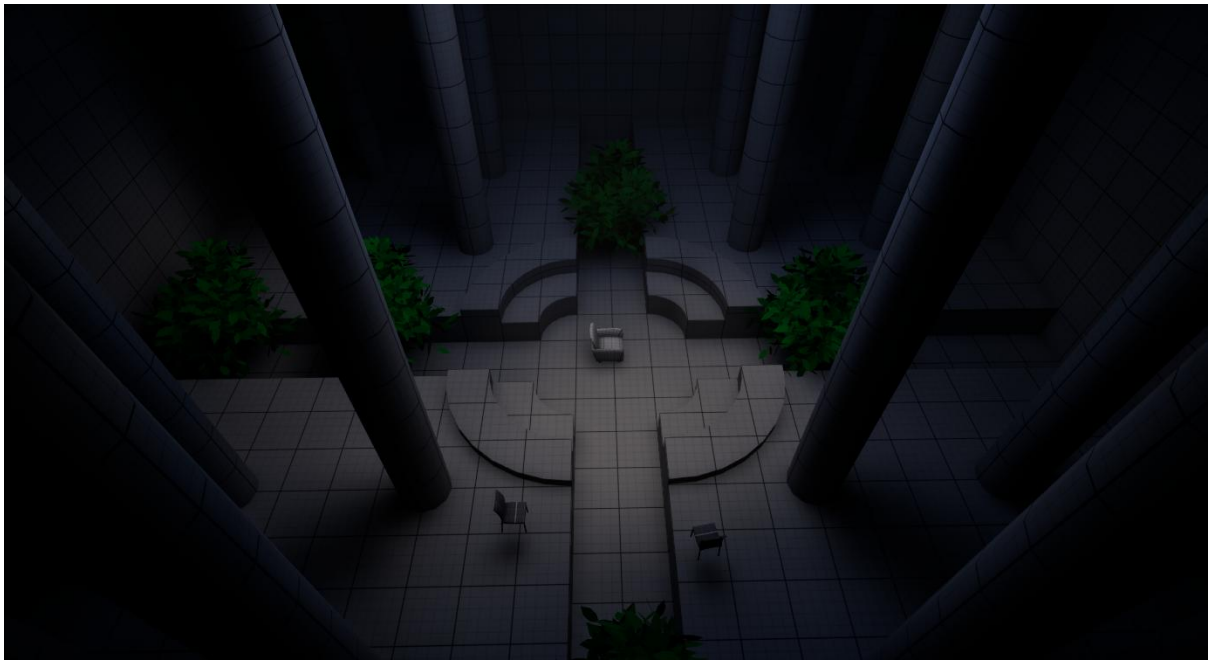
avoir. Ici, il y a la paix, la foi, la sécurité. À travers le vitrail, tu vois l'autre côté le puit central, le labyrinthe, la torture, le péché. Ici, tu es en sécurité. Il suffit de nous choisir.
On observe même des prisonniers agenouillés, déjà convertis. L'espace est pensé pour inspirer le **khouchou' (خشوع)**, un terme d'origine arabe, issu de la tradition islamique. Il désigne un état de **recueillement profond**, de **soumission du cœur**, d'humilité sincère face au divin. C'est une attitude intérieure où l'on ressent à la fois la paix, la peur respectueuse, la concentration et la présence spirituelle.

L'architecture en hauteur, les arches, les dômes : tout est conçu pour manipuler, pour créer l'illusion d'une religion lumineuse.

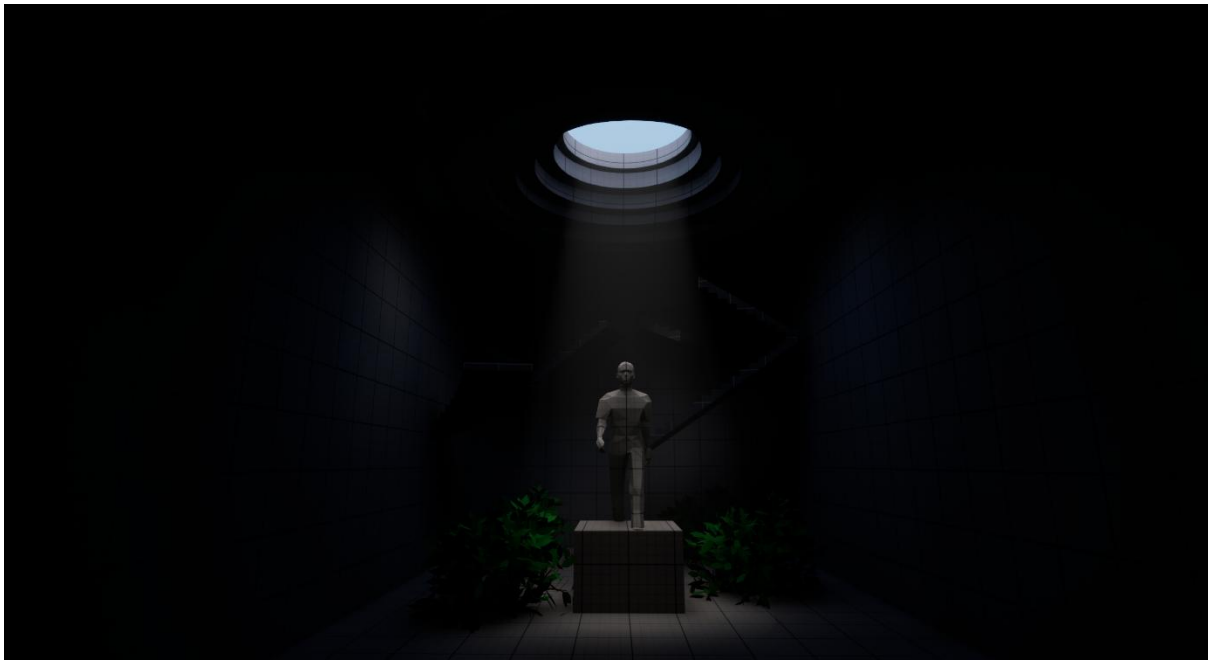
Mais ce "paradis" n'est qu'un décor. Au début, c'est une respiration, un souffle, une pause qui annonce qu'on se rapproche de la fin. Mais cet espace, qui semblait être un refuge, se transforme en torture. On finit par le détester, parce qu'on y retourne sans cesse. On y passe, on y repasse, encore et encore. On comprend alors qu'on est coincé dans une boucle, et que cette paix n'était qu'une illusion. Sa beauté devient un nouveau type de torture : une promesse de salut inaccessible. On comprend que l'église n'est pas la vérité, mais un outil de **conversion** et de **manipulation**.

La vérité absolue :

Puis vient la **troisième salle**, le cœur secret du lore : une rupture totale. Ici, pas de vitraux ni de pierres polies. On découvre alors une salle qui reprend la même architecture que la salle de torture, mais transformée. Pas de sang, pas de chaînes : à la place, des plantes qui poussent entre les pierres, de la terre, de la lumière naturelle.



Au centre, une simple chaise, comme un lieu de parole ou de prédication, mais sans violence, sans menace. Tout évoque **l'innocence** et le **message originel**, simple et pur. La salle de torture fut donc créée comme moquerie de cette salle. Ensuite, on trouve une statue entourée d'un petit jardin, de la terre, des plantes réelles, de la nature qui pousse seule et des escaliers en **géométrie non-euclidienne** qui suggèrent un espace divin, hors des lois humaines, un lieu pur et intact que la corruption n'a jamais atteint.



Ce lieu révèle que la religion originelle était pure, bienveillante, que le message n'a jamais prêché la violence. C'est le culte qui l'a corrompu pour justifier ses tortures, ses prisons, sa haine des "hérétiques".

Cette salle est volontairement inaccessible. Elle n'existe pas sur le chemin construit par le culte. On ne devait jamais la voir. On ne la trouve qu'en approchant de la fin. C'est là qu'on aperçoit des prisonniers avec des pioches, couverts de poussière, creusant sans relâche. Ils murmurent : *« Menteurs… il faut trouver le mensonge… la vérité… »* Elle a été cachée, volontairement effacée, parce qu'elle contredit tout ce que la prison veut imposer. Nous n'étions pas censés la trouver. Ce n'est pas un espace construit pour convertir, mais un témoignage silencieux de la vraie foi, dissimulée sous les mensonges.

Certains prisonniers ont percé la pierre, ouvert un trou dans le mur, et c'est par cette fissure que l'on découvre la salle cachée, révélée par la résistance et la désobéissance.

Cette brèche devient une sortie, une échappée vers la vérité, loin de la version imposée par le culte.

Ainsi, la progression environnementale devient un **discours visuel** :

1. **Torture** : punition, peur, culpabilité
2. **Église illusoire** : manipulation, conversion, faux salut
3. **Jardin-statue** : vérité, pureté, révélation

III – Level Design et Gabarits

Notre labyrinthe est construit sur plusieurs étages. Pour garder une logique modulaire et cohérente, nous sommes partis sur un **système de blocs en grille**.

Chaque bloc fait **12x12 mètres** et **6 mètres de hauteur** dans Unreal Engine 5 : cela crée une géométrie simple, mesurable, et répétable. Le centre de chaque face du carré peut avoir une ouverture qui donne vers un autre bloc, ce qui permet de composer en assemblant les blocs en grille.

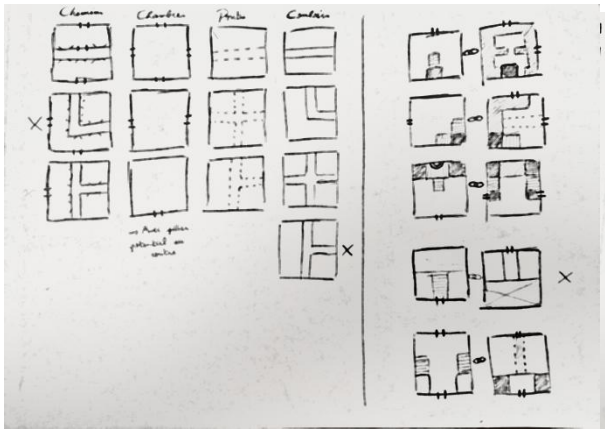
Nous avons choisi une grille de **8x8** blocs afin de garantir une bonne densité : suffisamment de chemins, de croisements, de cul-de-sacs et de boucles pour créer un véritable labyrinthe.

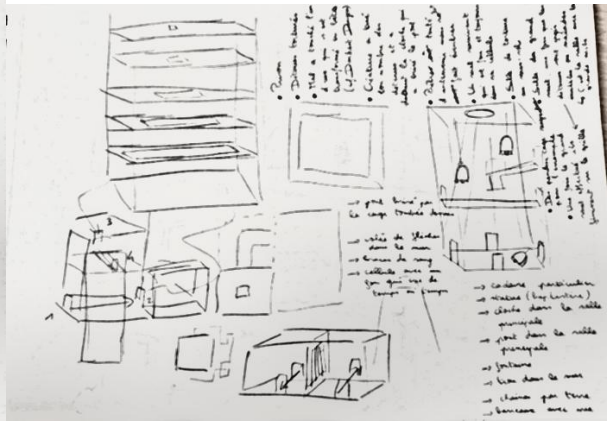
Mais nous voulions aussi un sentiment de verticalité, avec un puits central immense et oppressant. Pour que cette échelle soit crédible, il nous fallait au moins 30 étages de hauteur. Cela crée cependant un nombre colossal de blocs à générer.

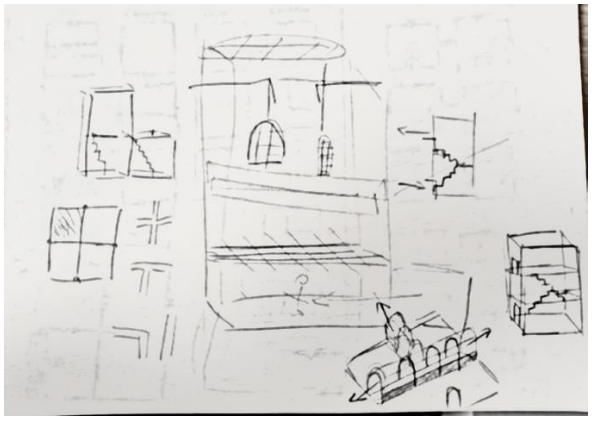
La solution était donc de ne pas utiliser toutes les cases de la grille ni tous les étages. Certains niveaux sont vides ou moins dense, et certains escaliers permettent de sauter un étage entier.

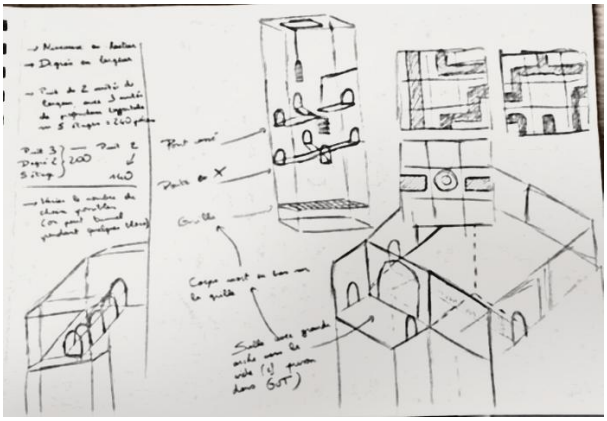
Cette stratégie conserve la densité du labyrinthe, tout en évitant de construire 30 × 64 blocs complets.

Voici le résultat de nos premiers croquis et essais :

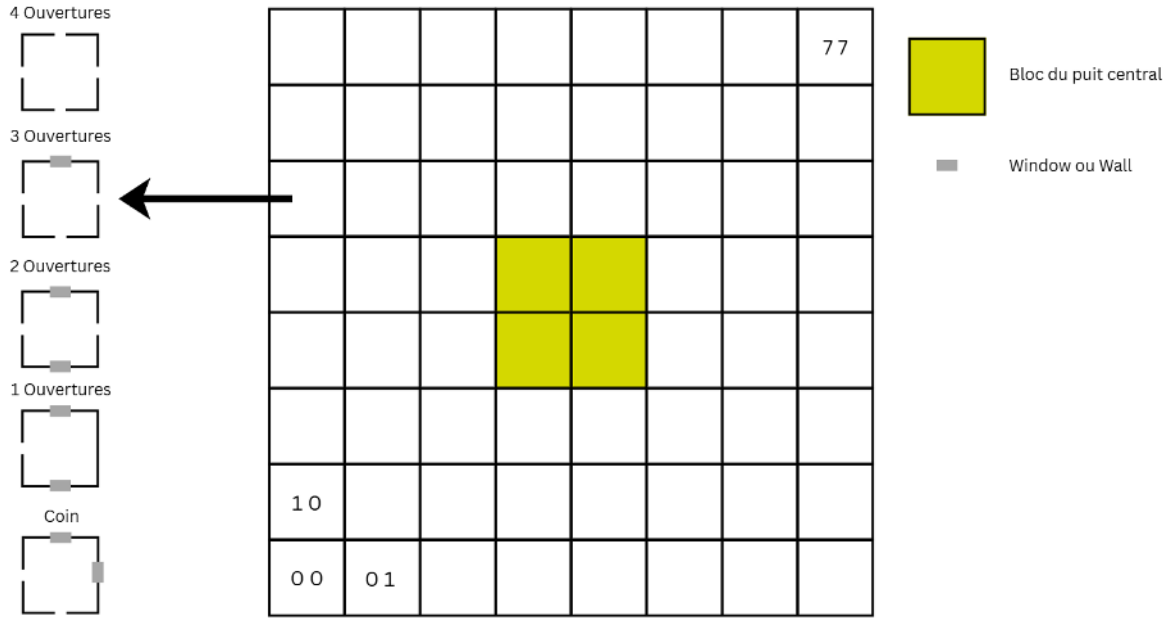








On est donc parti sur cette grille qui se répète sur 30 étages :

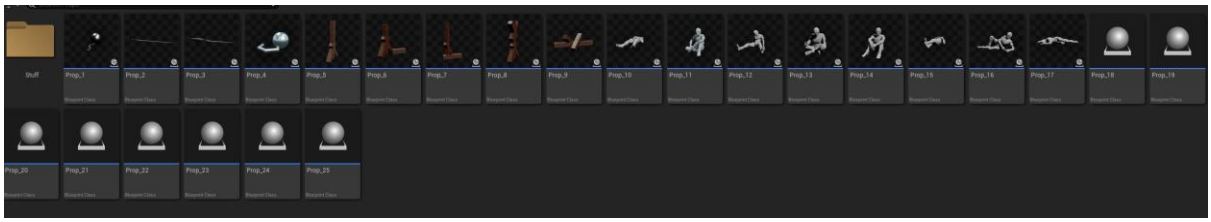


Nous avons donc conçu plusieurs familles de blocs : certains possèdent **1, 2, 3 ou 4 ouvertures**, d'autres sont des **blocs de coin**. Pour passer d'un étage à l'autre, nous utilisons des **blocs "Stairs"** qui font 12 mètres de hauteur (montée classique), ou 18 mètres pour les escaliers qui permettent de **sauter un étage** d'un coup.

À l'intérieur de chaque type de bloc, nous avons créé plusieurs variations architecturales : couloirs fermés, salles ouvertes, arches et ponts (utilisés pour le puit central ou à l'intérieur pour pouvoir visualiser un bloc en dessous).

On voulait que le joueur ait constamment la sensation de voir d'autres blocs, que tout est connecté, même s'il ne peut pas les atteindre, ou qu'il reconnaisse un espace déjà traversé sans pouvoir y retourner. Les blocs **"Windows"** sont essentiels à cette idée : ils ouvrent des vues entre les blocs ou vers le puit central. Certaines fenêtres sont des ouvertures qui donnent directement sur le vide du puit central, inspirées des prisons suspendues de l'Eyrie dans *Game of Thrones*. On voit le vide, on voit la chute. Le suicide, ou la liberté, semble toujours à portée de main.

Finalement, on a des props qui spawn dans les blocs aléatoirement :



Le labyrinthe entier est donc un **assemblage modulaire** de ces blocs sur une grille verticale et horizontale.

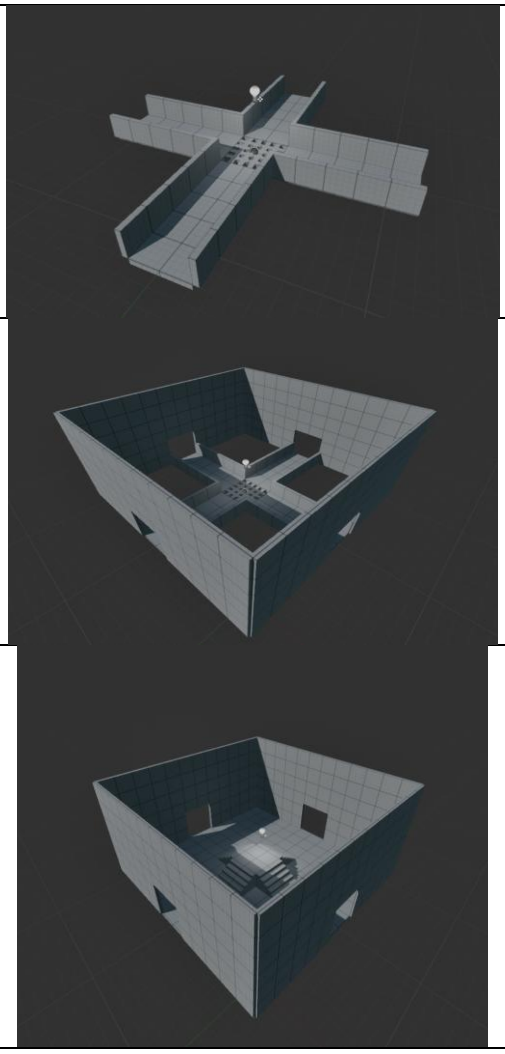
Types de blocs :

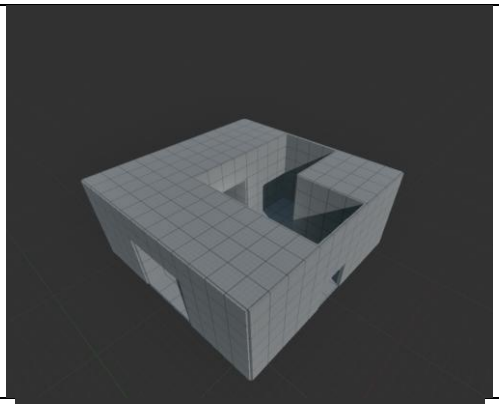
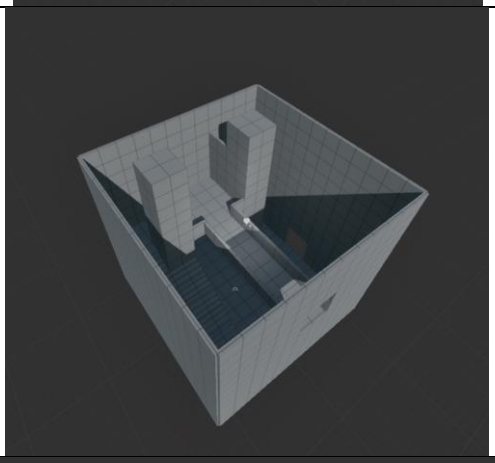
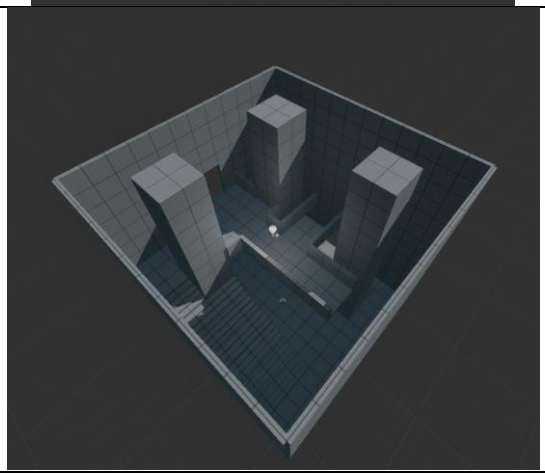
- Blocs à 1 ouverture
- Blocs à 2 ouvertures
- Blocs à 3 ouvertures

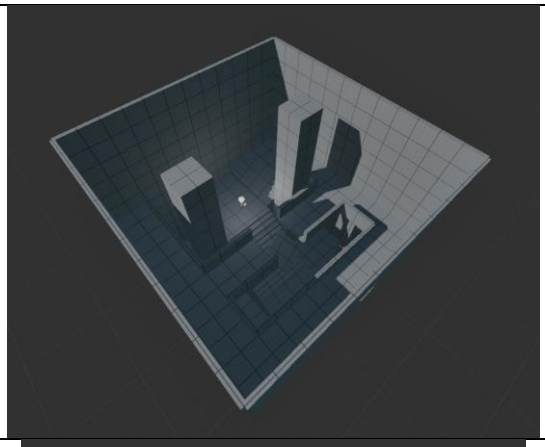
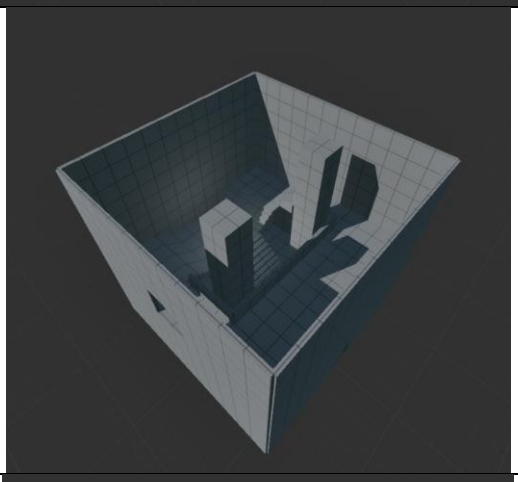
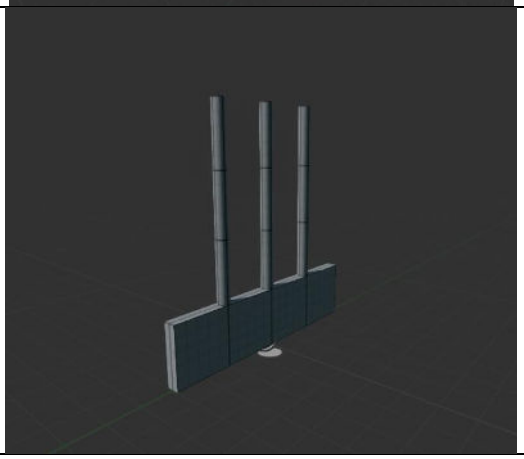
- Blocs à 4 ouvertures
- Blocs de coin
- Blocs Stairs (12m)
- Blocs Stairs Double (18m)
- Windows
- Props

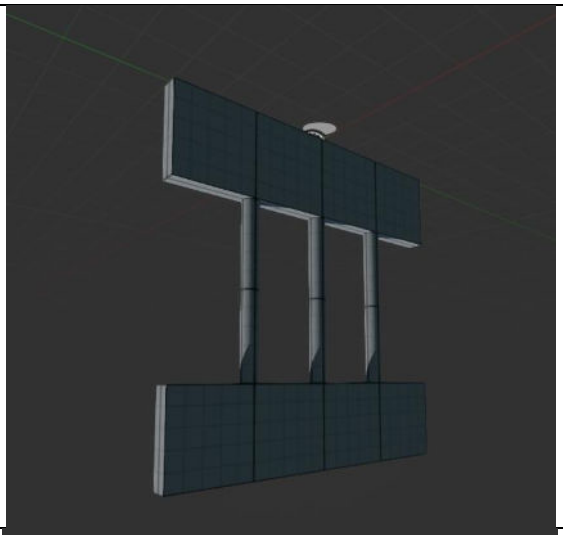
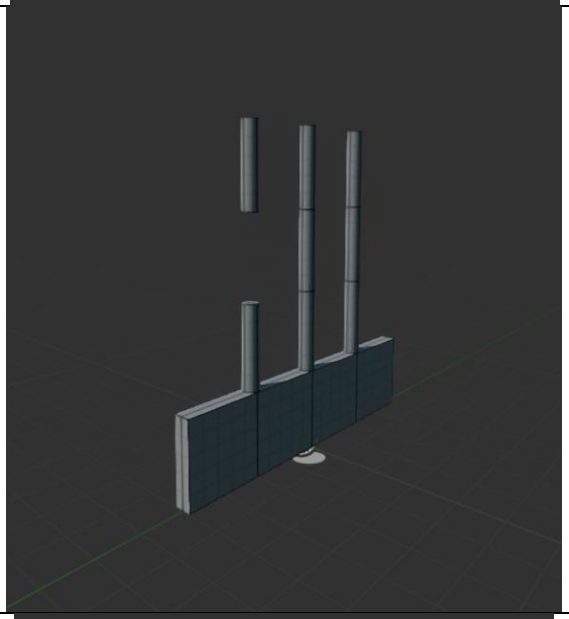
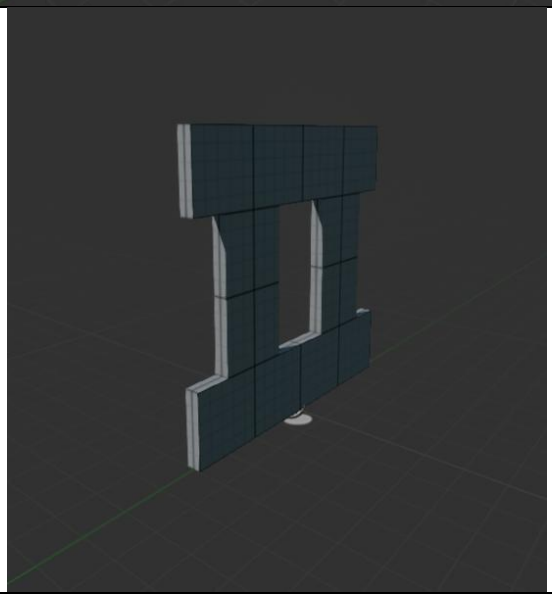
Variations de blocs :

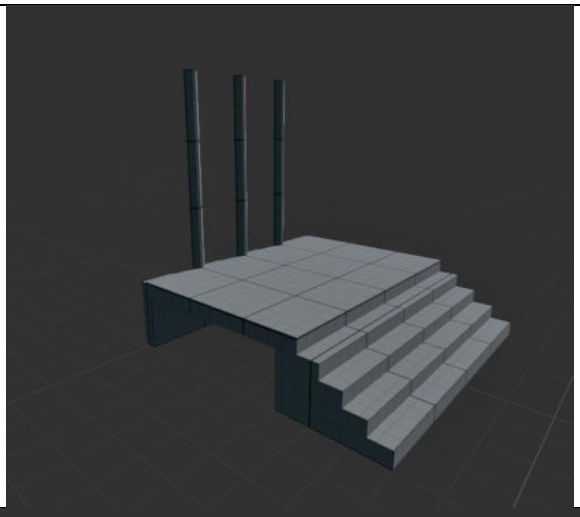
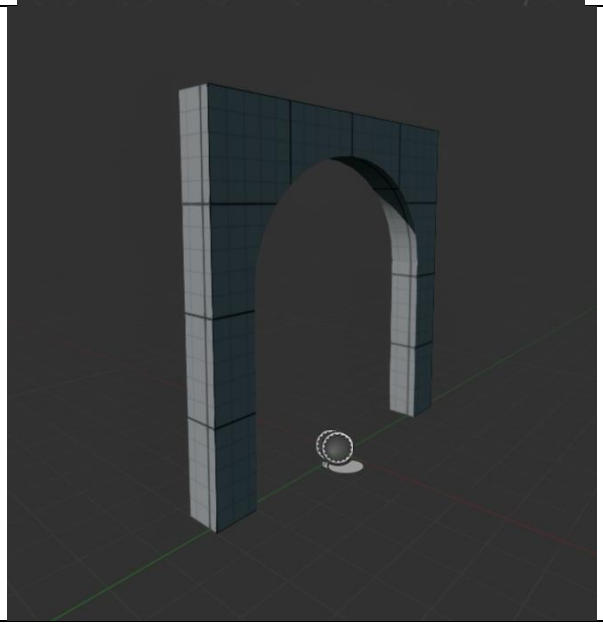
Pour chaque variation, on a une version pour chaque type. Par example, pour la variation Bridge Outside, on a Bridge Outside Corner, Bridge Outside 1 Door, Bridge Outside 2 Doors…

Variation du bloc	Module dans Unreal
Bridge Outside	
Bridge Inside	
Chamber	

Corridor	
Arch Path	
Stair G	
Stair U	

Stair EE		
Stair E		
Window 1		

Window 2		
Window 3		
Window 4		

Window 5		
Window Prison		

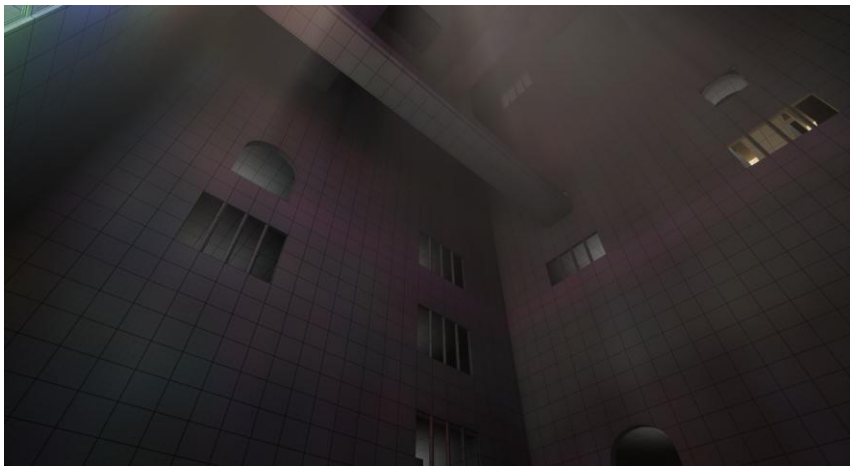
IV – Repérage et Guidage

Nous avons mis en place **2 niveaux de guidage** permettant au joueur de s'orienter dans le labyrinthe :

- un **guidage global**, lié à la progression générale dans la structure
- un **guidage local**, lié à la reconnaissance de chaque bloc.

Guidage global : comprendre où l'on se trouve dans le labyrinthe

Le **puit central** est le repère principal. Les ponts, les fenêtres et les plateformes du puit visibles à travers les fenêtres permettent de visualiser la hauteur atteinte. Aussi, plus on monte, plus l'espace est éclairé par la lumière venue d'en haut : la verticalité devient un langage.



Nous avons également 3 **salles spéciales** qui servent de points d'étape et de repères narratifs dans la progression.

Certains blocs n'apparaissent qu'à des étages précis (par exemple tous les 5 étages), ce qui permet au joueur de comprendre implicitement : *"Je suis à ce niveau du labyrinthe."*

Les blocs **Stairs** jouent aussi ce rôle. En prenant un escalier, le joueur a l'impression d'avancer. Mais ce repère peut être trompeur : certains embranchements comportent plusieurs escaliers qui ne mènent pas au bon étage.

Enfin, dès le début, on apprend que **progresser ne veut pas dire seulement monter**. La séquence initiale des étages est volontairement : **0 → -1 → 0 → 1**. Cela enseigne au joueur que descendre est parfois nécessaire pour remonter, et que la verticalité n'est pas une simple montée linéaire.

Guidage local : reconnaître un bloc, un espace, un endroit déjà exploré

Comme certains blocs se répètent, nous avons ajouté des **variations uniques** pour aider le joueur à se repérer à plus petite échelle. Par exemple :

- des corps,
- des éléments de torture,
- des tâches de sang,
- des gravures

Ces objets sont uniques et placés aléatoirement (mais jamais 2 fois identiques).

Chaque élément existe en plusieurs variations, ce qui évite la répétition visuelle tout en permettant la reconnaissance.

Les **fenêtres** jouent aussi un rôle local : elles offrent des vues uniques sur le puit central ou sur d'autres blocs, transformant chaque salle en espace mémorisable, tout comme les blocs ponts intérieurs qui donnent vers le bloc en dessous.

Enfin, la **combinaison des blocs** sert de repère : par exemple, le joueur peut se souvenir d'un enchaînement comme *« bloc 3 portes → bloc couloir → bloc arch path»* et reconnaître qu'il est déjà passé par là.

IV – Génération Procédurale

Nous utilisons un système de coordonnées **(x, y, z)** pour placer les blocs sur une grille 3D. La génération se fait en plusieurs étapes : création de la séquence d'étages, construction du chemin principal, génération des embranchements, puis habillage et variation des blocs.

1. Génération de la séquence d'étages

Avant de placer les blocs, nous générons une **séquence d'étages** allant du sol jusqu'à l'étage final définit par une variable (par exemple, 0 → 30). Cette séquence respecte plusieurs règles :

- La séquence commence **toujours** par : **0 → -1 → 0** (pour apprendre dès le début que descendre peut être nécessaire pour progresser).
- Une variable **respiration step** crée des étages « respirations » : on **saute** un étage (ex : on passe de 2 à 4).
- Une variable **down/up step** crée des aller-retour verticaux : on descend puis on remonte (ex : 5 → 4 → 5).
- Une variable pour le nombre d'étages

Cela produit une séquence du type :

0, -1, 0, 1, 2, 4, 5, 6, 8, 9, 10, 9, 10, 11, 13… jusqu'à 30.

Cette séquence servira de plan vertical pour traverser le labyrinthe.

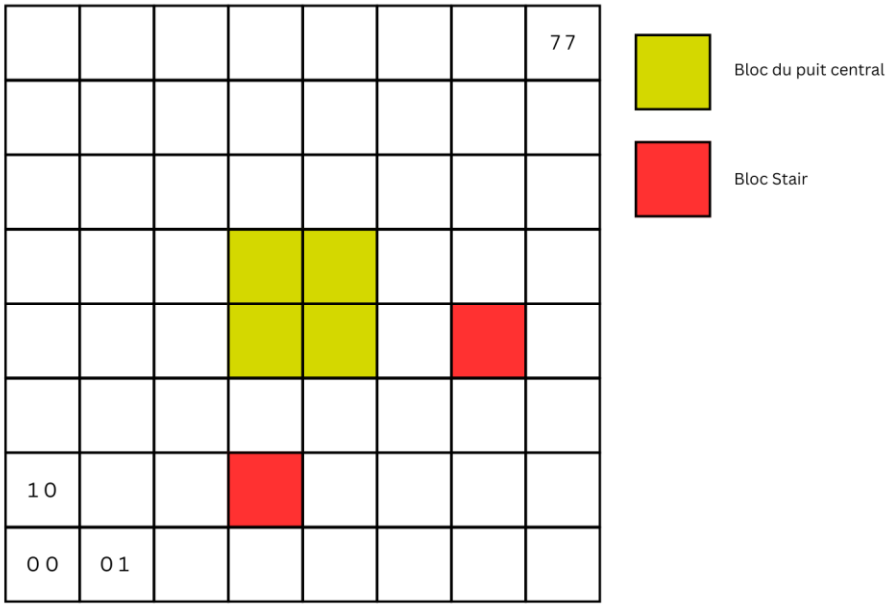
2. Génération du chemin principal

Point de départ : **coordonnée (2, 3, 0)**, connectée directement au puit central pour avoir l'effet de voir le puit au tout début.

Pour chaque étage de la séquence :

Étape 2.1 – Choix de la position des escaliers

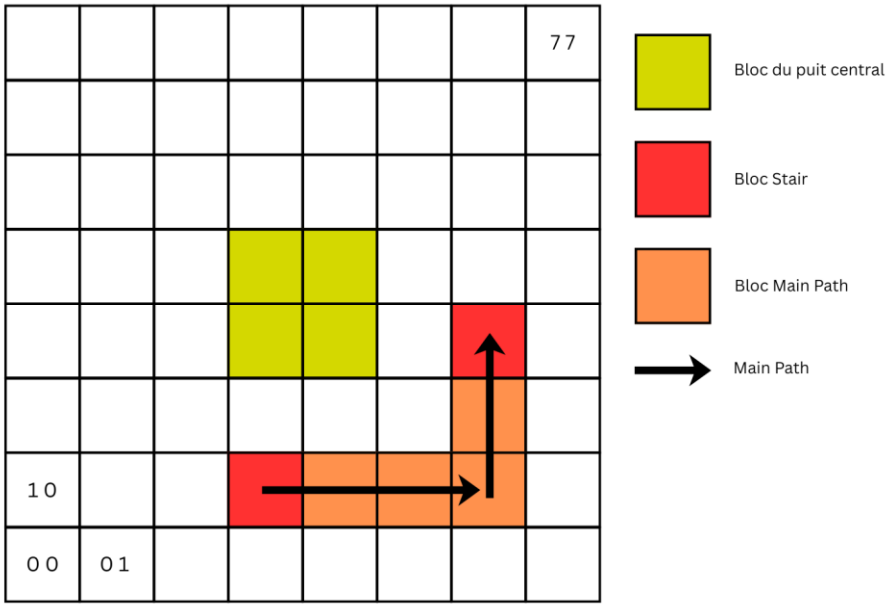
- On choisit la position du bloc "stair" sur l'étage courant.
- Le choix respecte une **distance minimale** rapport à la position précédente du bloc d'arrivé sur cet étage.
- Les blocs du puit central ne sont jamais choisis comme escaliers.



Étape 2.2 – Chemin le plus court jusqu'à l'escalier

On relie la position actuelle à la prochaine position d'escalier en calculant le **shortest path** sur la grille.
(Un algorithme de type **A*** ou **Dijkstra 2D**)

Ce chemin devient un segment du **chemin principal**.



Depuis l'escalier, on monte (ou descend) vers l'étage suivant indiqué par la séquence, et on répète ce processus pour chaque étage en prenant comme bloc de départ le bloc d'arrivé de l'étage précédent.

Ainsi se construit tout le chemin principal du labyrinthe, du début jusqu'au dernier étage.

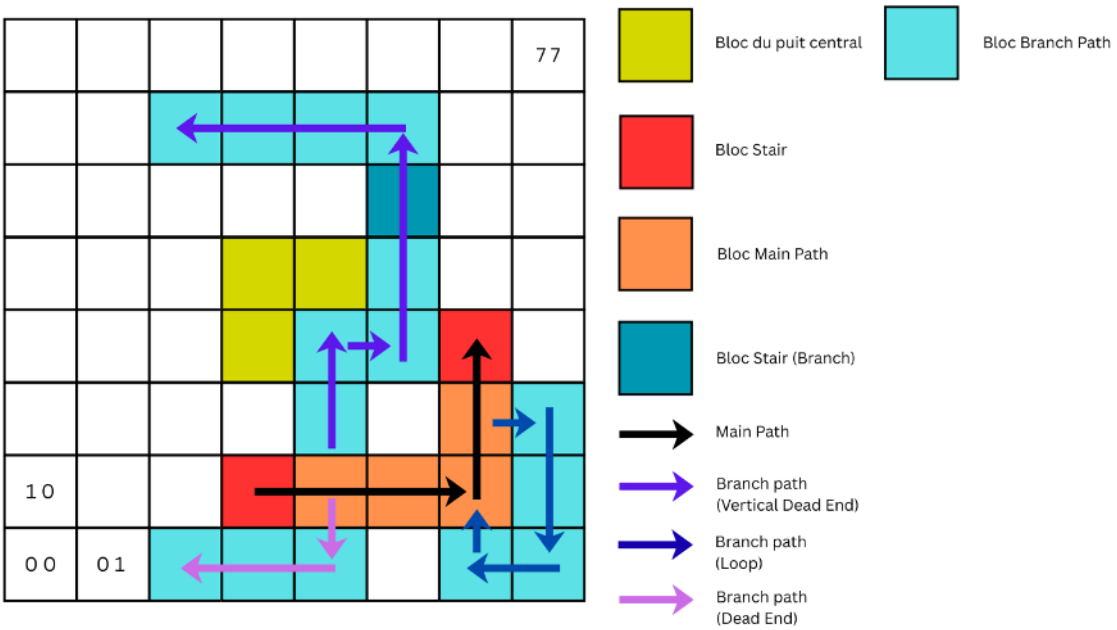
3. Génération des embranchements

Après le chemin principal, on ajoute des embranchements secondaires, sans jamais :

- réutiliser un bloc du chemin principal,
- ni traverser le chemin principal.

On génère 3 types :

1. **Dead End**
    - Un cul-de-sac simple partant du chemin principal.
2. **Loop**
    - Une boucle qui démarre et revient sur le chemin principal.
3. **Vertical Dead End**
    - Un cul-de-sac qui contient un escalier, permettant de monter ou descendre d'un étage, puis revenir dans une impasse.



4. Détermination du type de bloc et rotation

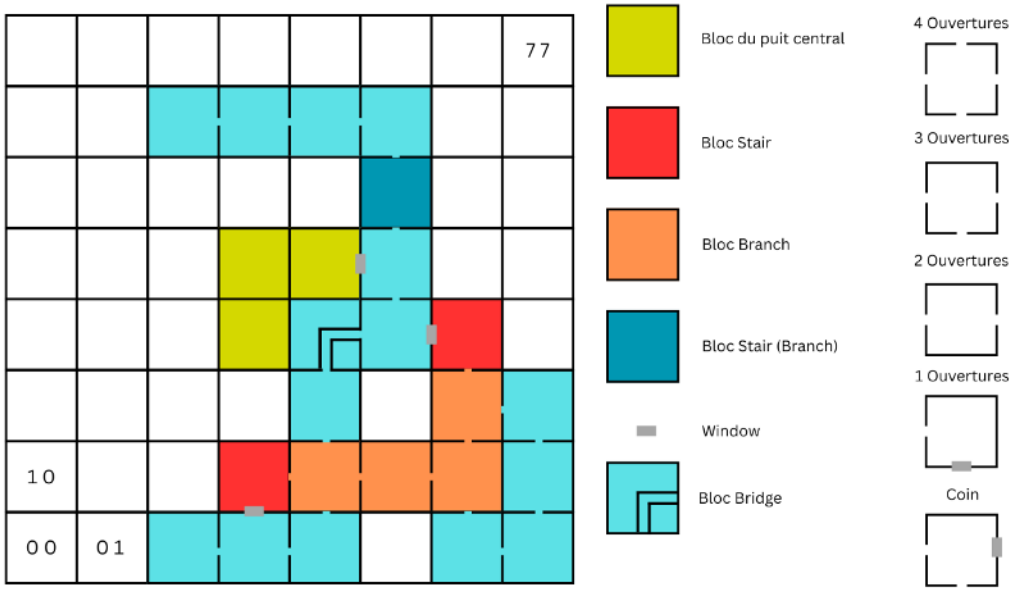
Pour chaque coordonnée de bloc, on identifie :

- Le nombre de voisins connectés
- la position des voisins (top, bottom, right, left)

Cela permet de choisir automatiquement le type de bloc et sa rotation :

- **Bloc 1 porte**
- **Bloc 2 portes**
- **Bloc 3 portes**
- **Bloc 4 portes**
- **Bloc de coin**

Par example, un bloc avec 2 voisins bottom et right signifie un bloc coin avec une rotation de 90 degrés afin que les ouvertures soient alignées. Les gabarits ont leur **pivot au centre**, ce qui permet une rotation propre.



Une fois le type défini, on sélectionne une **variation** (corridor, arch path, chambre...) selon une **probabilité** définie pour chaque bloc.

Règles particulières :

- Si le bloc touche le **puit central** → il devient automatiquement un **bloc pont.**

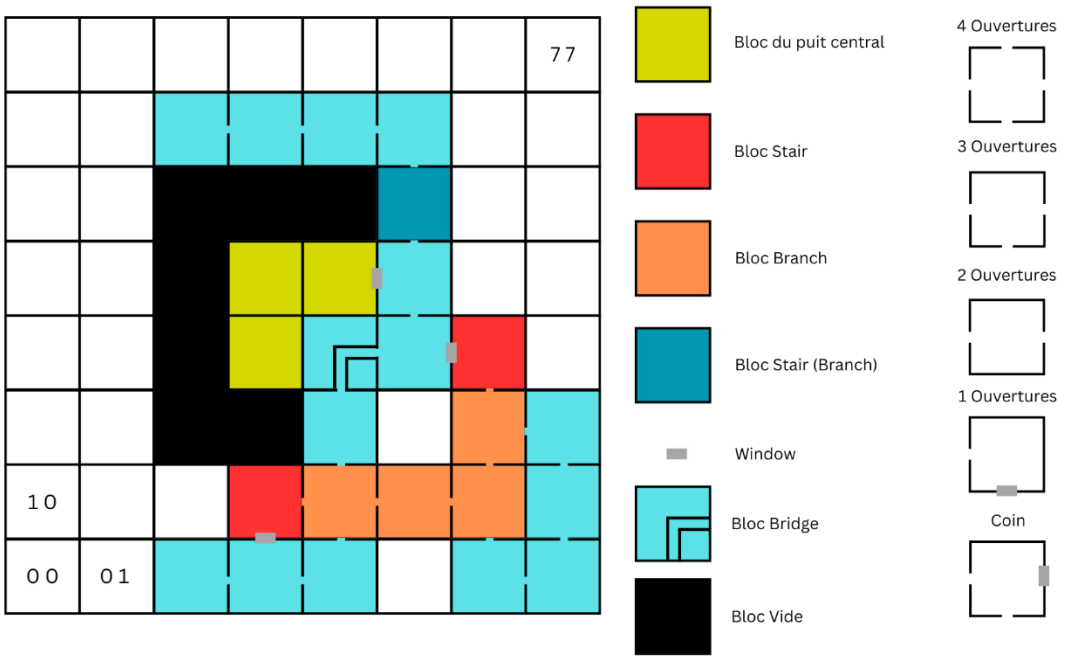
- Si c'est un pont intérieur → le bloc situé en dessous n'a pas de toit (on voit à travers).

Pour chaque face d'un bloc :

- Si le voisin est adjacent **mais non connecté**, on peut générer un **mur** ou une **fenêtre** (probabilité configurée).
- Si la face donne sur le **puit central** → c'est automatiquement une fenêtre pour offrir une vue verticale sur la hauteur.

7. Création du puits central et blocs inutilisés

Les blocs adjacents au puits central qui ne sont pas utilisés deviennent des **blocs vides**, fermés par des murs, pour créer la colonne verticale du puits.



8. Placement des props

Chaque bloc possède des placeholders prédéfinis.

On choisit des props (corps, vases, chaînes, os, cages, objets religieux…) dans une liste, **sans répétition**, et on les place dans ces placeholders.

9. Gestion des salles spéciales

Certaines salles sont **obligatoires** dans la narration (salle de torture, salle-église, salle de vérité, etc.).

Chaque salle définie des coordonnées pour ses ouvertures. Lorsque le chemin principal atteint l'étage de l'une de ces salles, l'algorithme **force** le chemin à passer à travers elle, avant de rejoindre l'escalier suivant.

V – Difficulté et Paramètres

Nous avons également défini un **système de difficulté progressive** réparti sur les étages du labyrinthe. Les niveaux **0 à 10** correspondent à une zone **facile**, les niveaux **10 à 20** à une zone **moyenne**, et les niveaux **20 à 30** à une zone **difficile**.

Chaque niveau de difficulté modifie plusieurs paramètres de génération : la longueur du chemin principal (donc la distance entre 2 escaliers), le nombre d'embranchements (dead ends, loops, vertical dead ends) ainsi que la longueur de ces branches. Plus on monte, plus ces valeurs augmentent : le labyrinthe devient **plus dense**, plus long et plus confus. La progression fonctionne comme un entonnoir inversé.

On a choisi ces valeurs :

Variable	**Easy**	**Medium**	**Hard**
Stair Distance	4	6	8
Dead End Count Range	0 – 3	1 – 4	2 – 5
Dead End Length Range	1 – 4	3 – 6	4 – 7
Loop Count Range	0 – 2	1 – 3	2 – 4
Loop Length Range	4 – 6	3 – 6	4 – 7
Vertical Count Range	0 – 2	1 – 3	2 – 4
Vertical Length Range	3 – 5	4 – 6	5 – 7

Les probabilités d'apparition des blocs suivent la logique suivante :

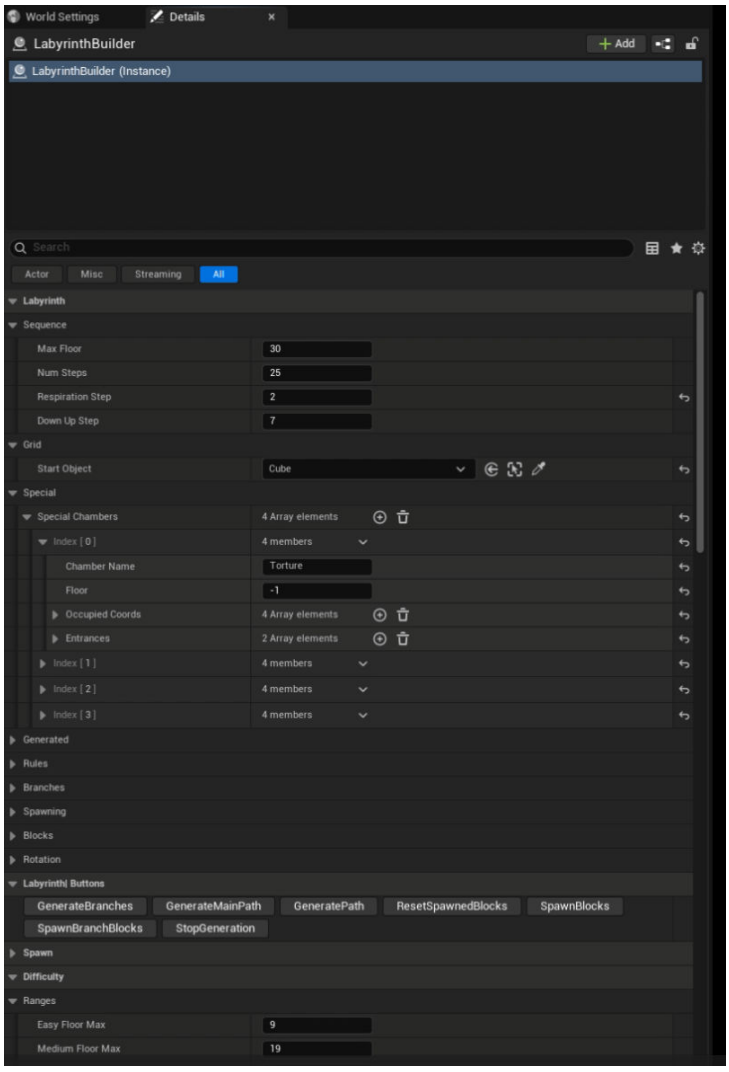
- les **corridors**, nos blocs les plus basiques, sont les plus fréquents (50%)
- les **chambers** sont moins probables (30%)
- les **bridges** sont les plus rares, ce qui les rend plus marquants et souvent associés à une progression (20%)

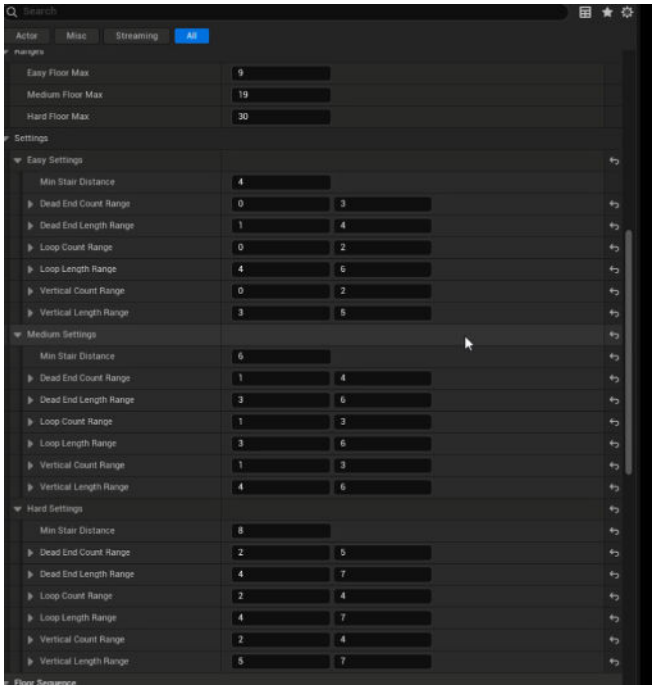
Les blocs **Arch Path** apparaissent plus sur les étages multiples de 5, ce qui sert également de repère visuel et de marque de progression pour le joueur.

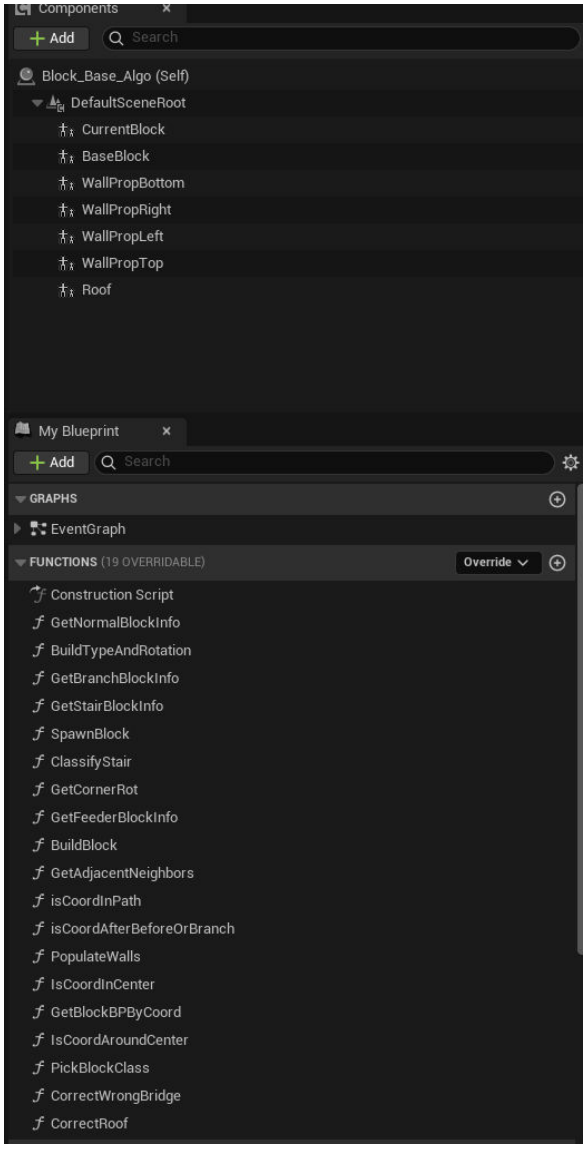
VI – Algorithme de Génération

Nous avons développé l'algorithme procédural en **C++,** notamment pour la génération de la séquence d'étages, le calcul des plus courts chemins et la construction de la grille du labyrinthe. Le C++ gère également le **spawn des blocs** sous forme de placeholders : chaque bloc est généré comme un **Actor Blueprint** auquel on attribue ses coordonnées (x, y, z).

Ensuite, l'**habillage** s'effectue côté Blueprint. À ce stade, chaque bloc analyse ses voisins, calcule sa **rotation** et choisit son **type** (1, 2, 3, 4 portes, coin, pont, etc.). Les variations internes (corridor, arche, chambre…) sont sélectionnées selon des **probabilités**, tout comme la génération éventuelle de **fenêtres** vers d'autres blocs ou vers le puit central. Cette combinaison C++ et Blueprint permet d'obtenir un système performant, entièrement automatisé et visuellement cohérent. On peut donc tester plusieurs variations du labyrinthe en modifiant les paramètres ci-dessous :




Search
Actor
Misc
Streaming
All
Easy Floor Max 9
Medium Floor Max 19
Hard Floor Max 30
Settings
Easy Settings
Min Stair Distance 4
Dead End Count Range 0 3
Dead End Length Range 1 4
Loop Count Range 0 2
Loop Length Range 4 6
Vertical Count Range 0 2
Vertical Length Range 3 5
Medium Settings
Min Stair Distance 6
Dead End Count Range 1 4
Dead End Length Range 3 6
Loop Count Range 1 3
Loop Length Range 3 6
Vertical Count Range 1 3
Vertical Length Range 4 6
Hard Settings
Min Stair Distance 8
Dead End Count Range 2 5
Dead End Length Range 4 7
Loop Count Range 2 4
Loop Length Range 4 7
Vertical Count Range 2 4
Vertical Length Range 5 7


Components
Add
Search
Block_Base_Algo (Self)
DefaultSceneRoot
CurrentBlock
BaseBlock
WallPropBottom
WallPropRight
WallPropLeft
WallPropTop
Roof
My Blueprint
Add
Search
GRAPHS
EventGraph
FUNCTIONS (19 OVERRIDABLE)
Override
Construction Script
GetNormalBlockInfo
BuildTypeAndRotation
GetBranchBlockInfo
GetStairBlockInfo
SpawnBlock
ClassifyStair
GetCornerRot
GetFeederBlockInfo
BuildBlock
GetAdjacentNeighbors
isCoordInPath
isCoordAfterBeforeOrBranch
PopulateWalls
IsCoordInCenter
GetBlockBPByCoord
IsCoordAroundCenter
PickBlockClass
CorrectWrongBridge
CorrectRoof