Projet Tower Defense

Objectif : Réaliser intégralement un jeu de type Tower-Defense en utilisant le moteur Unity.

Part I

Présentation du projet

1 Introduction

Le tower defense (souvent abrégée en TD) est un type de jeu vidéo où l'objectif est de défendre une zone contre des vagues successives d'ennemis se déplaçant suivant un itinéraire ou non, en construisant et en améliorant progressivement des tours défensives [1].

2 Présentation du projet

Votre groupe est chargé, au sein d'une entreprise de jeu vidéo, de créer un prototype permettant, d'une part, le chargement d'une carte produite par un éditeur externe, et d'autre part, la construction sur cette carte de différents batiments pour empêcher des vagues de monstres d'atteindre la sortie. Ce jeu sera nommé Tower Defense, mais la division marketing, peu inspirée, est prête à vous suivre pour tout nouveau nom.

Le projet sera réalisé par groupe de 3. Il sera nécéssaire de bien répartir le travail pour remplir tous les objectifs.

3 Travail à fournir

Prendre connaissance des fonctionnalités demandées dans le game design document (voir partie II).

Chargement et rendu d'une carte :

- Chargement et visualisation d'une carte à partir du dossier `Maps`.
- Création du graphe des chemins.
- Vérification de la jouabilité de la carte.
- Controlleur de caméra permettant de se déplacer sur la carte.

Développement du jeu :

- Gestion déroulement du jeu.
- Ajout, suppression de bâtiments et de tours.
- Spawn et déplacement des monstres.
- Gestion des actions des tours et bâtiments.
- Visualisation des tirs des tours.
- Gestion de l'energie (centrale et graphe de distribution).
- Menus pour pouvoir choisir une carte et lancer une partie.
- Système de paramêtres sauvegardés dans les PlayerPrefs Unity.

4 Planning Prévisionnel

- 2025-12-04 Départ du projet, formation des groupes
- 2025-12-16 Séance encadrée
- 2026-01-26 Séance encadrée
- 2026-01-29 Soutenances/Démonstration et livraison du projet Unity.

5 Rendu attendu

Rendu attendu pour le projet :

- Executable fonctionnel du jeu.
- Une carte dans le format attendu (png suivant les spécifications 6.1)
- Repository Git contenant le projet complet.
- Un rapport écrit.
- Démonstration du projet pendant le cours du 29 janvier 2026.

Note: Tout élément indiqué dans un encart *Bonus* n'est pas attendu dans le rendu final mais sera récompensé.

5.1 Rapport

Contenu attendu dans le rapport :

- Noms du trinôme,
- Introduction.
- Présentation de l'application / description détaillée des fonctionnalités de l'application.
- Description globale de l'architecture de votre application (justification des découpages, etc.).
- Répartition du travail (comment avez découpez le travail pour arriver a avancer en parallèle),
- Les succès / difficultés rencontrées
- "Résultats" obtenus par votre application (capture d'écran, temps de calcul...)
- Conclusion.

Part II

Game Design Document

Note: Le jeu devra être en 3D.

6 Carte

6.1 Format de fichier de carte à charger

Pour pouvoir permettre à la communauté de partager ses meilleures cartes, il a été décidé d'utiliser comme format de représentation une image en couleur au format png.

Une carte est composé de cases (ou tile) d'une taille de 1 unité Unity (1uu). Les cartes peuvent avoir des tailles différentes définit par la taille de la texture (1 pixel = 1 case).

Chaque pixel définit le type de la case correspondant à sa position en fonction de sa couleur. Les couleurs et leur signification sont des constantes décrite dans le tableau 1. Tout pixel ayant une couleur différente de celles décrites dans les spécifications appartient à la catégorie des zones constructibles.

Les chemins sont connectés via des nœuds d'intersection. Les chemins relient toujours ces noeuds de manière rectiligne. Ainsi tout coude dans un chemin doit être marqué par une intersection. Les zones d'entrée et de sortie sont également symbolisées par un noeud.

RGB	Type
(299, 229, 229)	Zone non constructible
(255, 233, 127)	Chemin
(255, 178, 127)	intersection
(0, 255, 33)	Entrée
(255, 0, 0)	Destination
other	Zone constructible

Table 1: Type de case en fonction de la couleur du pixel

6.2 Validation d'une carte

Plusieurs conditions doivent être remplie pour qu'une carte soit valide. Votre application devra être capable, lors du chargement d'une carte, de vérifier si une telle carte est valide ou non.

Voici les différentes exigences :

- Chaque chemin est rectiligne (horizontal ou vertical) et relie 2 Nœuds. (entrée, sortie ou intersection)
- Bonne validité des chemins : Le long d'un chemin entre 2 noeuds on doit rester sur le chemin (ne pas croiser d'autre pixel que des pixels de type chemin).
- Existence d'au moins une zone d'entrée et de sortie.
- Existence d'au moins un chemin entre la zone d'entrée et de sortie : c'est un parcours en profondeur simple du graphe des chemins.

6.3 Rendu de la carte

Le rendu de la carte devra se faire en utilisant un système de tiles 3D à instancier lors du chargement du fichier carte en fonction des règles décrite précédemment. Dans un premier temps, une approche simple associant une tile à un type de case pour être utilisée. Il pourra être nécéssaire dans un deuxième temps d'utiliser des variations en fonction des cases voisines pour afficher correctement certaines cases (virages, intersections, ...).

Vous pouvez utiliser le pack d'assets de votre choix pour le rendu du terrain (avec la contrainte qu'il doit utiliser des assets 3D). Exemple d'asset en reférence [2].

Bonus : Décorez la partie non jouable de la carte pour ne pas voir le fond gris d'Unity

6.4 Caméra

Le joueur à avoir la possibilité de déplacer la caméra à l'aide des touches Z,Q,S,D du clavier.

Bonus : Ajouter la possibilité de zoomer/dézoomer sur la caméra

Bonus : Ajouter un controle de la caméra à la souris, par drag and drop pour le déplacement et molette pour le zoom

7 Déroulement du jeu

Le joueur commence la partie avec une somme d'argent initiale et un certain nombre de points de vie. Il peut construire des tours de défense sur les zones constructibles de la carte. Ces tours sont là pour détruire des monstres, arrivant par vagues successives de plus en plus grosses, et qui cherchent à atteindre la zone de sortie. Si un monstre parvient à la zone de sortie, le joueur perd une quantité de points de vie et le monstre disparait. Lorsque le joueur arrive à 0 point de vie, la partie est perdue. Chaque monstre détruit par les tours rapporte de l'argent et des points au joueur. Cet argent permet au joueur de construire des nouvelles tours pour gérer les vagues suivantes. De même lorsqu'une vague se termine, un score bonus est attribué au joueur. Le mode de jeu principal sera un mode "survie", le but du jeu est d'arriver le plus loin possible et d'avoir le score le plus élevé possible.

L'ensemble des éléments du jeu est donc :

- La carte décrite avec ses chemins
- Les monstres
- Les tours
- Les batiments

Les bâtiments comprennent d'une part des centrales d'énergie qui alimentent les tours de défense et d'autres parts des installations qui apportent des améliorations aux tours. Si une tour n'est pas complètement alimentée, elle ne peut pas tirer.

Note : L'équilibrage est laissé à votre discretion. La note finale ne dépendra pas de la qualité de cette équilibrage, mais il est conseillé de travailler dessus pour fournir un prototype de qualité.

8 Les chemins

Les chemins de la carte forment un graphe que nous allons utiliser pour indiquer aux monstres leur chemin. Dans ce graphe, chaque sommet représente une intersection, la zone de sortie ou une zone d'entrée. Les arcs représentent les chemins rectilignes reliant intersections, zones d'entrée/sortie. Vous devrez donc construire un graphe représentant l'ensemble des chemins de la carte. Cela vous permettra entre autre de vérifier l'existence d'un chemin entre la/les zones d'entrée et celle de sortie pour la validation de la carte.

Les données associés aux sommets de ce graphe sont simplement le type de sommet (intersection, zones d'entrée/sortie) ainsi que les successeurs de ce sommet. Les arcs disposent également de données, notamment une valeur représentant le risque pour les monstres d'emprunter cette partie de chemin. Cette valeur est calculée comme suit :
$valarc = longueur \sum recouvrement$

longueur est le nombre de cases entre les 2 sommets du graphe. *recouvrement* indique, pour chaque tour de défense, le nombre de cases de l'arc situé dans la zone d'effet de la tour (voir section suivante). Cette pondération des arcs du graphe sera utile pour le calcul du chemin des monstres.

9 Elements de jeux

9.1 Les monstres

Les monstres disposent de points de vie et d'une résistance propre à chacun des types de tour (voir section suivante). Les monstres arrivent par groupes de plus en plus nombreux. Chaque monstre tué rapporte de l'argent au joueur. Vous devrez implémenter au moins 2 types différents de monstre ayant des caractéristiques distinctes (vitesse, resistance aux types de dégats, dégats aux joueur, ...).

Graphiquement, les monstres seront représentés dans le jeu par des modèles 3D animés. Vous pouvez utiliser les assets de votre choix (avec la contrainte qu'il soient en 3D avec des animations de course). Exemple d'asset en reférence [3].

9.1.1 Comportement des monstres

Si il y a plusieurs zone d'entrée, l'application choisira aléatoirement une zone d'entrée pour une vague de monstre donnée. Lorsqu'il arrive en jeu, tout monstre doit construire son trajet pour atteindre la sortie. Afin de rendre un peu plus malin nos monstres, nous allons utiliser le graphe de chemin (dont les arcs sont pondérés par leur distance et l'influence des tours). Chaque monstre devra alors lancer l'algorithme de plus court chemin sur le graphe de chemin pour rejoindre la zone de sortie. L'utilisation de Dijkstra [6] est indiquée. Ainsi chaque monstre connaîtra la liste des nœuds du graphe de chemin constituant son trajet. Entre chaque nœud le parcours est rectiligne.

Bonus : Avoir des comportements différents en fonction des monstres (plus sur, plus court, aléatoire, ...)

9.2 Les tours

Le joueur peut créer des tours de défense qui tireront automatiquement sur les monstres. Les tours ont quatres caractéristiques : la puissance, la portée, la cadence et la consommation d'énergie. La puissance indique les dégâts effectués par la tour. La portée indique en cases la distance à laquelle la tour peut tirer. La cadence indique en tir par secondes l'intervalle entre deux tirs. Les tours tirent sur le monstre le plus proche d'elle. La consommage d'énergie indique la quantité d'énergie nécessaire à la tour pour fonctionner (voir section 9.3.1).

Les tours sont de quatre types :

- Les tours de type rouge : Ces tours infligent beaucoup de dégâts mais ont une cadence de feu faible.
- Les tours de type vert : Elles tirent très rapidement mais ont une faible portée et occasionne des dommages moyens.
- Les tours de type jaune : Elles occasionnent peu de dégât, ont une portée très limité mais une bonne cadence de tir et tirent sur tous les monstres à leur portée.
- Les tours de type bleue : Elles ont une très bonne portée et une bonne cadence de tir mais occasionnent peu de dégâts.

Les différents paramètres de ces différentes tours (dégâts, portée, cadence) sont laissés à votre appréciation. Chaque tour a également un coût d'achat pour pouvoir être placée dans la carte. Les tours sont représentés par des modèles 3D et ont une taille de 1 case. Leur rayon d'action est calculé avec la distance de Manhattan [4].

De plus les tours ne peuvent être construites que sur une zone constructible (c'est à dire pas sur un chemin, ni sur une autre tour ou un autre bâtiment).

Bonus: Le choix des cibles des tours varie en fonction du type de tour (plus proche, plus éloigné, avec le moins de point de point de vie, ...)

9.2.1 Visualisation des tirs des tours

Pour que le joueur comprenne bien sur quels monstres tirent les tours, il faudra mettre en place une visualisation des projectiles lancés par les tours sur les monstres.

Bonus: Les projectiles et les trajectoires changent en fonction des types de tours

9.3 Les bâtiments

Il existe dans notre jeu, deux types distincts de bâtiment : les centrales et les installations. Graphiquement, les bâtiments sont également représentés comme les tours, par des modèles 3D à poser sur les cases.

Comme pour les tours, on doit veiller à ce que le placement des bâtiments soient valides c'est à dire ne recouvre pas ni un chemin, ni une tour.

9.3.1 Les centrales

Les centrales fournissent l'énergie aux tours de défense. En effet, sans énergie, les tours de défense ne tirent pas. On vous laisse le soin de déterminer le coût des centrales et la quantité d'énergie qu'elles produisent. Ces centrales transmettent leur énergie à toutes les tours situées dans leur rayon d'action. Comme pour les tours, ce rayon d'action est calculé avec la distance de Manhattan [4]. Toutes les tours dont au moins une case ce trouve dans la portée de la centrale sont ainsi alimentées.

De plus, les tours alimentées peuvent elle même servir de relai d'énergie pour toutes les tours situées à leur portée. Afin de gérer au mieux la distribution d'énergie, il convient donc de réaliser un graphe représentant les centrales (il peut évidemment y en avoir plusieurs), les tours et leur relation de portée. Ainsi chaque sommet de ce graphe représentera soit une centrale, soit une tour. Et un arc reliera deux sommets si la tour (ou la centrale) du sommet destination est à portée de la tour (ou de la centrale) du sommet d'origine. Pour connaître les tours qui sont alimentées, il convient de faire un parcours en largeur de ce graphe à partir de l'ensemble des centrales. Sur un même "niveau" dans le parcours en largeur, la répartition de l'énergie restante à distribuer est laissée à votre appréciation.

9.3.2 Les installations

Trois installations peuvent être construites par le joueur : le radar, l'usine d'armement, et le stock de munitions. Ces bâtiments sont eux aussi munis d'une portée. Le radar permet d'augmenter de 25% la portée de toutes les tours situées dans sa zone d'effet. L'usine d'armement permet d'augmenter de 25% la puissance des tours dans sa zone d'effet. Enfin, le stock de munitions permet d'augmenter de 25% la cadence de tir des tours sans sa zone d'effet. Le coût et la portée de ces installations est laissé à votre appréciation.

9.4 Résumé des structures de données à utiliser

Au delà de la définition de structures pour les tours, bâtiments, monstres... vous aurez besoin, pour l'implémentation de ce jeu, de deux graphes : le graphe des chemins et le graphe d'énergie (les liens énergétiques centrales/tours). Vous avez toute latitude sur le choix de l'implémentation du graphe des chemins. Notez par ailleurs qu'il s'agit d'un graphe non orienté.

10 Menus et navigations

10.1 Menu principal

Menu d'accueil lors du lancement du jeu. Il contient 3 boutons :

- Jouer (qui mène à l'écran de lancement de partie).
- Paramètres (qui mène à un menu de paramétrage).
- Quitter

10.2 Ecran de lancement de partie

Affiche la liste des cartes disponible. Il contient 2 boutons :

- Lancer partie (qui utilise la carte sélectionné pour lancer une nouvelle partie).
- Retour (qui revient au menu principal).

Les cartes sont chargées depuis le dossier `Maps` situé à coté de l'executable du jeu.

10.3 Paramètres

Ce menu doit contenir les paramètres suivants :

- Le volume des effets
- Le volume de la musique
- Le volume général
- La résolution du jeu

Ces paramêtres doivent être sauvegardés en utilisant les `PlayerPrefs` d'Unity [5].

10.4 Interface pendant la partie

L'interface pendant la partie doit contenir les informations suivante :

- Le numéro de vague et le score du joueur et le nombre de monstre restant de la vague.
- L'argent restant disponible
- Les tours et batiments que le joueur peut construire avec possibilité de les sélectionner pour les placer sur la carte.
- Possibilité de selectionner un batiment ou monstre pour afficher ses caractéristiques.
- Un moyen de quitter pour revenir au menu principal.
- Barre de vie au-dessus des unités qui sont endommagées.

Bonus : indication des dégâts infligés à chaque coup

10.5 Pop-up de fin de partie

Lorsque la partie se termine, affichez un pop-up qui récapitule les informations de la partie (numéro de vague atteint, score, statisques de partie, ...). Ce panneau doit aussi proposer les options suivantes :

- Retourner au menu
- Recommencer (qui redémare la partie avec la même carte).

Part III

Annexes

11 Conseils et remarques

- Ce sujet de projet constitue un cahier des charges de l'application. Tout changement à propos des spécifications du projet doit être validée au préalable.
- Le temps qui vous est imparti n'est pas de trop pour réaliser l'application. N'attendez pas le dernier moment pour commencer.
- Il est très important que vous réfléchissiez avant de commencer à coder aux principaux modules, algorithmes et aux principales structures de données que vous utiliserez pour votre application. Il faut également que vous vous répartissiez le travail et que vous déterminiez les tâches à réaliser en priorité.
- Ne rédigez pas le rapport à la dernière minute sinon il sera bâclé et cela se sent toujours.
- Le projet est à faire par trinôme. Il est impératif que chacun d'entre vous travaille sur une partie et non pas tous "en même temps" (plusieurs qui regarde un travailler). Sinon vous n'aurez pas le temps de tout faire.
- N'oubliez pas de tester votre application à chaque spécification implémentée. Il est impensable de tout coder puis de tout vérifier après. Pour les tests, confectionnez vous tout d'abord de petites cartes avec un chemin extrêmement simple. Si cela marche vous pouvez passer à plus gros ou plus complexe.

Tout outils, spécifications, fonctionnalités supplémentaires seront récompensés. On peut penser notamment à :

- Différents types de terrain pour les chemins ralentissant ou accélérant les monstres.
- Affichage de la zone d'effet d'une tour ou batiment sélectionné.
- Affichage du trajet prévu par un monstre sélectionné.
- Les tours peuvent avoir une emprunte au sol d'une forme spécifique (carré de 4 cases, barre de 3 cases, ...). Possibilité de tourner les tours pendant la phase de placement si l'emprunte est asymétrique.
- Système d'upgrade des tours et batiments
- ...

Néanmoins, si une nouvelle fonctionnalité modifiait même de manière légère les spécifications, notamment le format des fichiers de cartes, présentes dans cette description de projet alors cette nouvelle fonctionnalité devra être validée au préalable. Elle devra aussi être indiquée clairement dans le rapport.

References

[1] Wikipedia *Tower Defense* https://fr.wikipedia.org/wiki/Tower_defense

[2] Kenney *Tower Defense Kit* https://kenney.nl/assets/tower-defense-kit

[3] Unity Asset Store *Stylized Free Skeleton* https://assetstore.unity.com/packages/3d/characters/creatures/stylized-free-skeleton-298650

[4] Wikipedia *Distance de Manhattan* https://fr.wikipedia.org/wiki/Distance_de_Manhattan

[5] Unity *Documentation PlayerPrefs* https://docs.unity3d.com/6000.0/Documentation/ScriptReference/PlayerPrefs.html

[6] Unity *Dijkstra's Algorithm* https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm