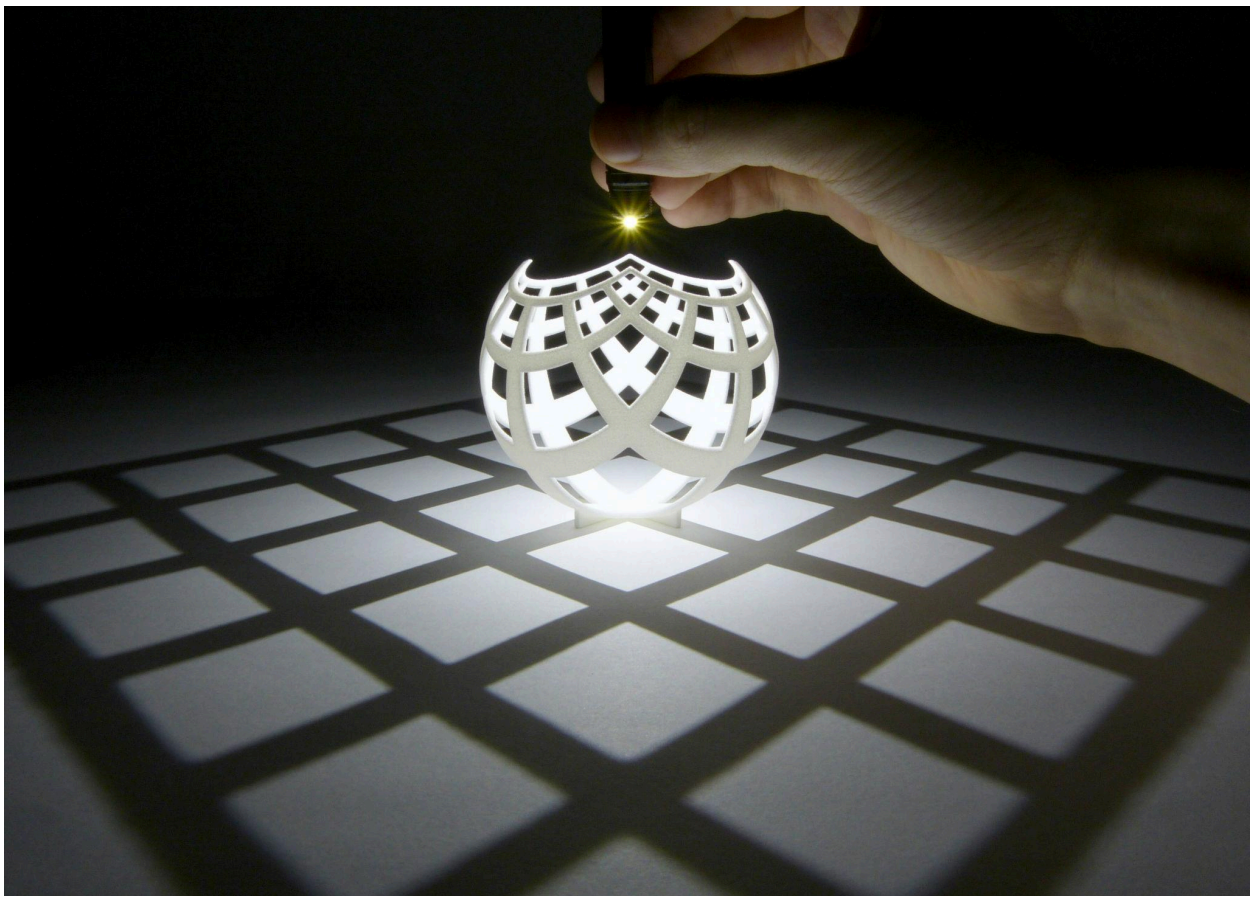
PROJET TUTORÉ

APPLICATION VR EN 4 DIMENSIONS
ÉTUDE PRÉALABLE



KOMODZINSKI Jawad
RYSAK Hugo
TROHA Stanislas

S5 RA-IL 1

Tuteur : DUPONT Laurent

26 Novembre 2024

Présentation du projet

Vision du produit

1. Qui va acheter/utiliser le produit ? Qui est la cible ?

Le produit cible des amateurs de réalité virtuelle et des passionnés de 4e dimensions. Cela va de l'étudiant souhaitant mieux comprendre les espaces multidimensionnels, aux chercheurs travaillant sur des concepts liés à la 4D, en passant par des enseignants voulant un outil pédagogique permettant de visualiser les dimensions de manière intuitive.

Les passionnés de VR pourront vivre une expérience unique liée à des concepts mathématiques.

2. À quels besoins le produit va-t-il répondre ?

Le produit répond à plusieurs besoins pédagogiques, professionnels et récréatifs.

- Comprendre les concepts abstraits de la 4D (Hypercube, rotations dans des plans 4D, etc.).
- Visualiser des objets 4D et les manipuler.
- Simplifier l'enseignement de ce sujet complexe.
- Permettre une exploration libre de l'espace 4D avec des intéractions intuitives en VR.

3. Quelles sont les fonctionnalités critiques pour répondre aux besoins de façon à avoir un produit réussi ?

Pour garantir le succès du produit, il est essentielles d'avoir :

- Une navigation fluide dans l'espace 4D.
- La possibilité de manipuler et visualiser les objets 4D.
- La possibilité de basculer entre les perspectives pour explorer différentes "coupes" ou projections.

4. Comment le produit se situe-t-il par rapport aux produits existants sur le marché?

Le produit est innovant car il vise une niche presque inexplorée, car il y a peu d'applications ou d'outils qui existent pour la visualisation 4D.

Notre application apporte une dimension immersive contrairement aux logiciels traditionnels qui projettent la 4D en 2D ou 3D de manière.

Aussi, les utilisateurs peuvent explorer plusieurs projections ou "coupes" de l'espace 4D en temps réel, ce qui rendra le tout plus compréhensible.

Etude Technique :

Translations en 4D :

Une **translation** est une transformation géométrique qui déplace chaque point d'un espace selon un **vecteur donné**, sans changer la forme ni l'orientation de l'objet. En 4D, la translation suit le même principe qu'en 2D ou 3D.

Un point **(x, y, z, w)** est déplacé selon un vecteur de translation **(tx, ty, tz, tw)**. Le point translaté **(x', y', z', w')** est donné par :

$$x' = x + t_x, \quad y' = y + t_y, \quad z' = z + t_z, \quad w' = w + t_w$$

En notation matricielle (vecteurs colonnes) :

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \\ t_w \end{bmatrix}$$

Quelques propriétés de la translation :

1. Si deux translations successives sont appliquées, leur ordre n'a pas d'importance :

$$T_1 + T_2 = T_2 + T_1$$

2. Les distances entre les points restent identiques après une translation.

Rotation en espace 4D :

En espace 4D, la rotation est plus complexe qu'en 2D ou 3D, car il existe **6 plans** possibles de rotation, correspondant aux **combinaisons d'axes** parmi les quatre dimensions (x , y, z, w).

Les 6 plans de rotation :

En 4D, une rotation se produit dans un plan défini par deux axes. Les 6 plans possibles sont :

XY, **XZ**, **XW**, **YZ**, **YW**, **ZW**

Matrice de rotation pour un plan donné :

Chaque rotation dans un plan est définie par un angle **θ**, et les coordonnées des points changent en conséquence. Les axes qui ne participent pas à la rotation restent **inchangés**. Voici les matrices associées :

1. Rotation dans le plan **XY** :

$$R_{XY} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2. Rotation dans le plan **XW** :

$$R_{XW} = \begin{bmatrix} \cos\theta & 0 & 0 & -\sin\theta \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \sin\theta & 0 & 0 & \cos\theta \end{bmatrix}$$

3. Rotation dans le plan **ZW** :

$$R_{ZW} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\theta & -\sin\theta \\ 0 & 0 & \sin\theta & \cos\theta \end{bmatrix}$$

Les matrices pour les autres plans (XZ, YZ, YW) suivent un schéma similaire.

Exemple :

Considérons un point **(1, 0, 0, 0)** en 4D. Si on applique une rotation de **90°** (θ = π/2) dans le plan **XW**, la matrice de rotation est :

$$R_{XW} = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Le point transformé devient :

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = R_{XW} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Le point **(1, 0, 0, 0)** a été déplacé dans la quatrième dimension pour devenir **(0, 0, 0, 1)**.

Transformation combinée :

Pour appliquer **plusieurs** rotations dans différents plans, on **multiplie** leurs matrices respectives. Par exemple, pour une rotation dans les plans XY et XW, on calcule :

$$R = R_{XY} \cdot R_{XW}$$

Cela permet de **combiner** les rotations en une seule transformation.

Quelques propriétés de la rotation :

1. Les matrices de rotation en 4D sont **orthogonales**, c'est-à-dire que leur **transposée** est leur **inverse** :
$$R^\top = R^{-1}.$$
2. La distance entre les points est conservée après une rotation.
3. Une rotation dans le plan XY par un angle θ peut être **annulée** par une rotation inverse (-θ).

Projections 4D en 3D :

La projection est essentielle pour **visualiser** des objets 4D dans notre espace 3D, tout comme nous projetons des objets 3D sur une surface 2D pour les afficher sur un écran. En 4D, un objet est représenté par ses coordonnées **(x, y, z, w)**, et nous devons le réduire en une représentation 3D **(x', y', z')**.

Les différentes projections :

1. Projection orthogonale

La projection orthogonale consiste à "**ignorer**" une des dimensions. On projette l'objet 4D **directement** sur un espace 3D selon un axe spécifique.

Si l'axe **w** est ignoré :

$$x' = x, \quad y' = y, \quad z' = z$$

Le point 4D **(x, y, z, w)** devient simplement **(x, y, z)** en 3D.

Cette projection est simple mais ne montre pas la **profondeur** dans la 4e dimension, ce qui la rend peu **intuitive** pour la visualisation.

2. Projection perspective

La projection perspective est une méthode courante pour représenter des objets en 3D sur une surface plane (en 2D), où les objets plus éloignés apparaissent plus petits et ceux proches de l'observateur apparaissent plus grands. Ce principe est utilisé pour créer une représentation réaliste de l'espace sur une surface.

La projection perspective en 4D suit un principe similaire à la projection en 3D, mais avec une dimension supplémentaire. Dans un espace 4D, les objets ont quatre coordonnées : **(x, y, z, w)**.

L'idée est de projeter ces objets en 4D vers un espace 3D en choisissant **un point de fuite** dans l'espace 4D et en appliquant une **transformation** qui réduit la quatrième dimension.

Exemple :

Soit un point **P = (x, y, z, w)** dans l'espace 4D. Pour une projection vers un hyperplan **w = 0** (ce qui peut être considéré comme un espace 3D visible depuis un observateur dans 4D), la projection peut être formulée comme suit :

$$x' = \frac{x \cdot d}{w + d}, \quad y' = \frac{y \cdot d}{w + d}, \quad z' = \frac{z \cdot d}{w + d}$$

- **x'**, **y'** et **z'** sont les coordonnées **projetées** dans l'espace 3D.
- **d** est la distance de l'observateur à l'hyperplan **w = 0**. C'est la **distance entre le plan de projection** (où l'objet 4D sera "rendu" en 3D) et le **point de vue ou la caméra virtuelle** (l'observateur). On peut calculer de manière dynamique ou alors en choisir un fixe.

Cette formule utilise une **proportionnalité inverse** pour la projection, ce qui signifie que les objets plus proches de l'observateur (avec un w plus faible) apparaîtront plus grands, et les objets plus éloignés (avec un w plus grand) apparaîtront plus petits.

En d'autres termes, si nous avons un hypercube 4D, la projection de ses différentes faces dépendra de la valeur de **w** pour chaque point de l'hypercube. Les faces les plus proches de l'observateur en 4D (plus petites valeurs de **w**) apparaîtront plus grandes dans la projection, tandis que les faces les plus éloignées (valeurs de **w** plus grandes) seront réduites.

En 4D, la projection perspective est essentielle pour obtenir une **représentation visuelle** réaliste d'objets tels que des hypercubes, des hypersphères, etc. Elle permet de donner une "**profondeur**" à l'objet projeté, permettant de mieux comprendre ses structures internes, même si nous ne pouvons pas directement percevoir la quatrième dimension.

Sans une projection perspective, il serait difficile de rendre compte de la structure 4D dans notre espace 3D, car tout apparaîtrait trop abstrait ou plat. La projection perspective 4D nous aide à visualiser ces objets de manière qu'ils semblent réalistes, tout en étant conscients de leur nature multidimensionnelle.

3. Projection stéréographique

La projection stéréographique est une technique utilisée pour projeter des points d'une hypersphère 4D sur un espace 3D. Elle est couramment employée en mathématiques et en visualisation, car elle **préserve les angles** et offre une représentation intuitive des structures 4D.

Contrairement à la projection perspective, la projection stéréographique n'introduit pas de point de fuite. Elle est particulièrement utile pour projeter **des structures sphériques**, comme les **hypersphères**.

L'idée principale est de "**projeter**" chaque point d'une hypersphère 4D (ou d'un espace 4D) sur un hyperplan 3D **tangent** à la sphère. La projection se fait depuis un "**pôle**" de la sphère, similaire à la projection stéréographique en 3D.

Il y a **3 éléments clés** :

- **L'espace source**, l'hypersphère 4D centrée à l'origine.
- **L'espace cible**, l'hyperplan 3D tangent à la sphère.
- **Le pôle de projection**, le point spécifique sur la sphère, souvent (0, 0, 0, R), où R est le rayon de la sphère.

Définition du pôle :

Le pôle est un **point spécifique** sur une sphère ou une hypersphère (dans le cas 4D) qui sert de point de **référence** pour effectuer une projection stéréographique. En d'autres termes, c'est le point à partir duquel les lignes de projection sont tracées.

Pour une hypersphère 4D centrée à l'origine, le pôle est généralement choisi comme l'un des points les plus **extrêmes** dans une direction donnée.

Par exemple, si l'hypersphère a un rayon R, **le pôle nord** est **(0, 0, 0, R)**. Donc **le pôle sud** est **(0, 0, 0, -R)**.

Exemple :

Supposons que nous ayons une hypersphère 4D de rayon R centrée à l'origine, c'est-à-dire définie par l'équation :

$$x^2 + y^2 + z^2 + w^2 = R^2$$

La projection stéréographique part **du pôle nord (0, 0, 0, R)** et projette un point **P = (x, y, z, w)** sur l'hyperplan **w = -R**.

Donc, la projection d'un point **P** vers un point **P' = (x', y', z')** se fait selon la formule :

$$x' = \frac{x}{1 - \frac{w}{R}}, \quad y' = \frac{y}{1 - \frac{w}{R}}, \quad z' = \frac{z}{1 - \frac{w}{R}}$$

Si on choisit **le pôle sud (0, 0, 0, -R)**, la projection se fait vers un hyperplan différent **w = R** (au lieu de **w = -R**). Donc la formule devient :

$$x' = \frac{x}{1 + \frac{w}{R}}, \quad y' = \frac{y}{1 + \frac{w}{R}}, \quad z' = \frac{z}{1 + \frac{w}{R}}$$

Donc le pôle définit **le signe de R**.

Pour se le représenter, il faut imaginer une hypersphère 4D centrée à l'origine. La projection stéréographique consiste à tracer une ligne droite entre un point P de la sphère et un pôle donné (par exemple, le point (0, 0, 0, R)). Le point projeté est l'intersection de cette ligne avec le plan hyperplan w = -R.

Cette méthode conserve les angles entre courbes. En revanche, les distances ne sont pas préservées, mais les objets proches du pôle sont fortement agrandis tandis que ceux proches du plan hyperplan w = -R apparaissent plus naturels.

Par exemple, pour une hypersphère, la projection stéréographique donne des sphères concentriques en 3D (des sphères qui ont le même centre). Cela représente une "coupure" dans l'espace 4D, chaque niveau de w correspondant à une sphère 3D différente.

Un hypercube peut également être projeté par stéréographie pour créer une image déformée mais reconnaissable d'un cube imbriqué. Les arêtes convergent de manière plus uniforme que dans une projection perspective.

A noter :

Il faut bien comprendre que les projections permettent de représenter un objet 4D en 3D, mais il faut souvent animer l'objet (par rotation ou changement d'angle de perspective) pour bien comprendre sa structure.

Fonctionnalités du système

Phase 1 : Création et manipulation des objets 4D

- Création de formes géométriques 4D courantes (Hypercubes, Hypersphères, Pentatopes).
- Visualisation de ces formes géométriques 4D dans une dimension 3D.
- Projection sous différentes vues.
- Manipulation de ces objets (rotation, translation).
- Afficher plusieurs instance de l'objet pour le voir sous plusieurs angles
- Créer une interface utilisateur pour sélectionner les différentes propriétés voulues.

Phase 1.2 : Légère implémentation VR

- Découverte des contrôles/commandes VR.
- Interface adaptée à la VR.
- Implémentation de la base de l'application.

Phase 2 : Déplacement dans un univers 4D

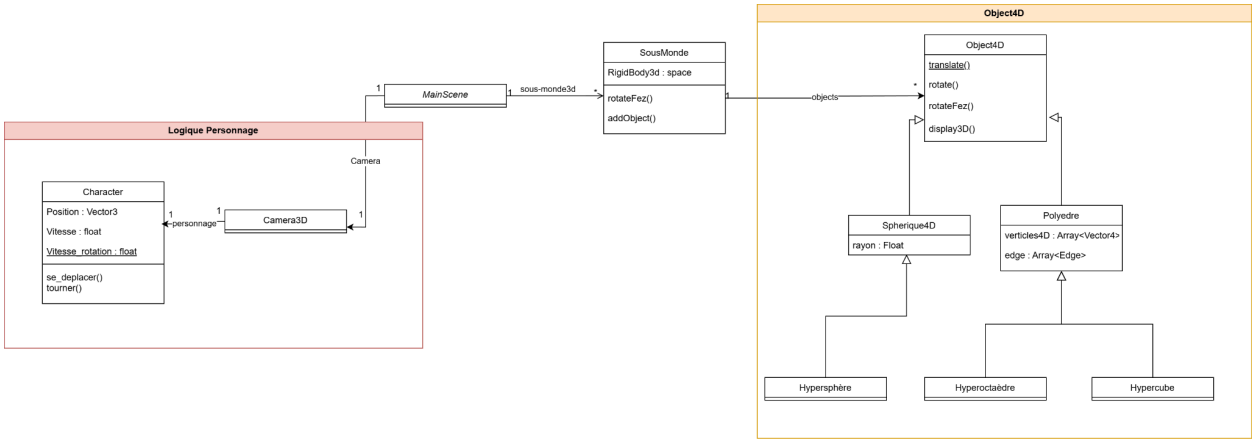
- Déplacement à la "Fez".
- Rotation de l'hyperespace.
- Découvrir des passages secrets grâce à l'interface Fez.

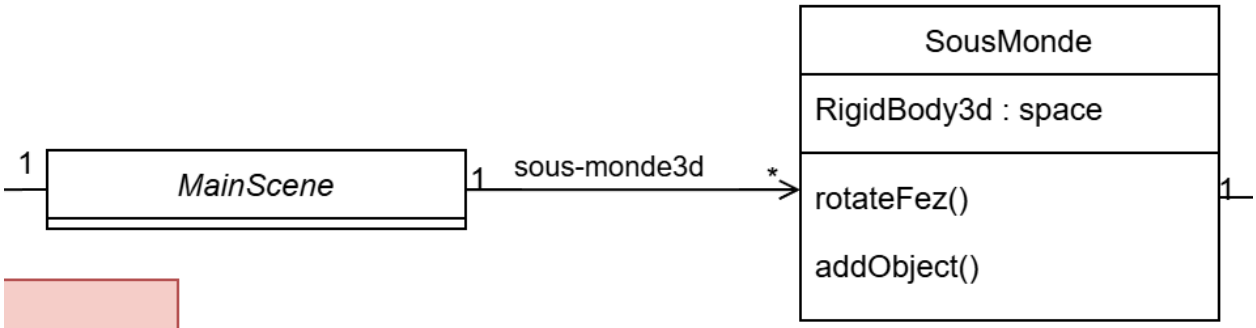
Phase 3 : Implémentation en VR

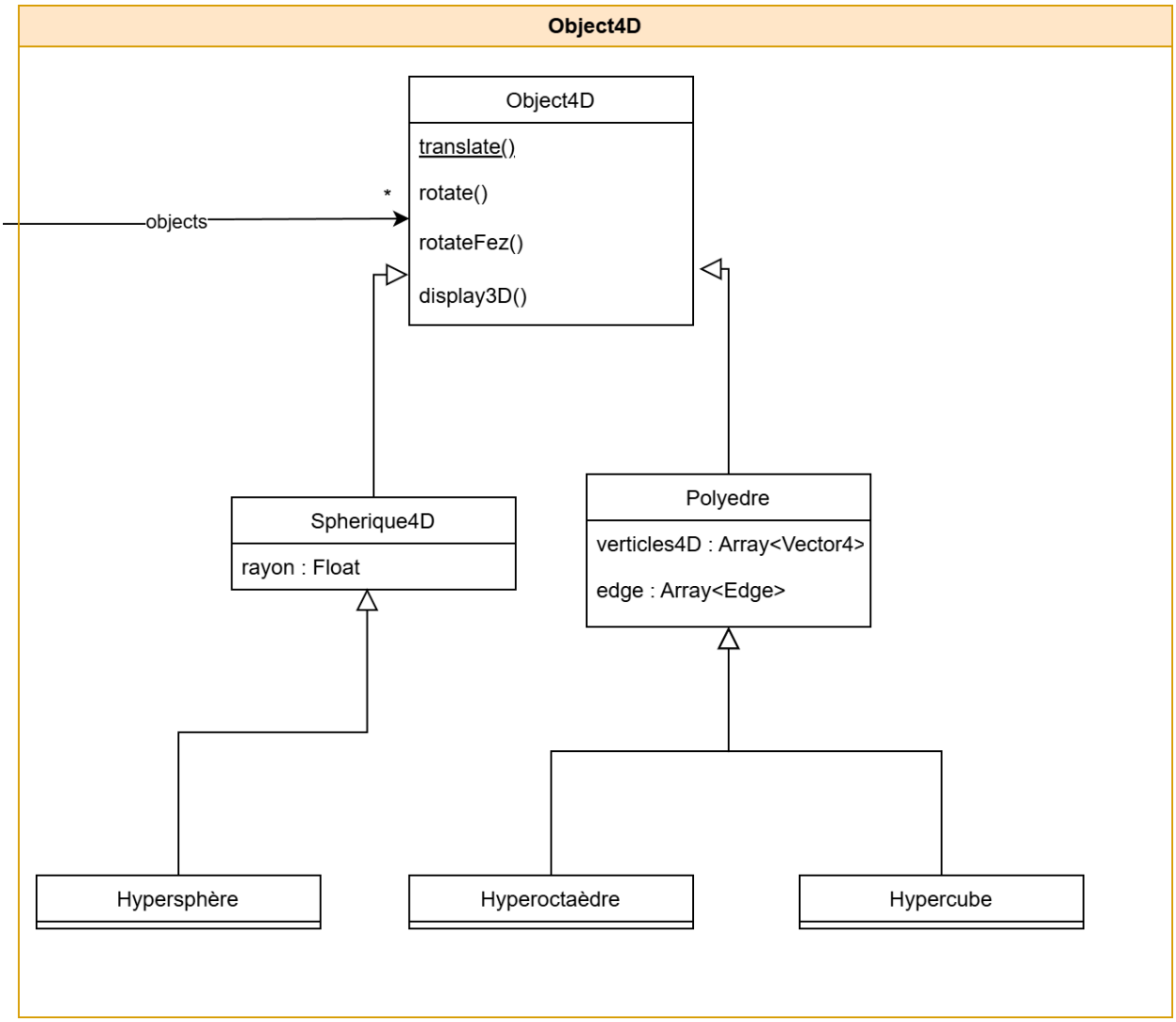
- Adaptation des commandes de manipulations d'objets.
- Adaptation des commandes de navigation dans l'hyperespace.

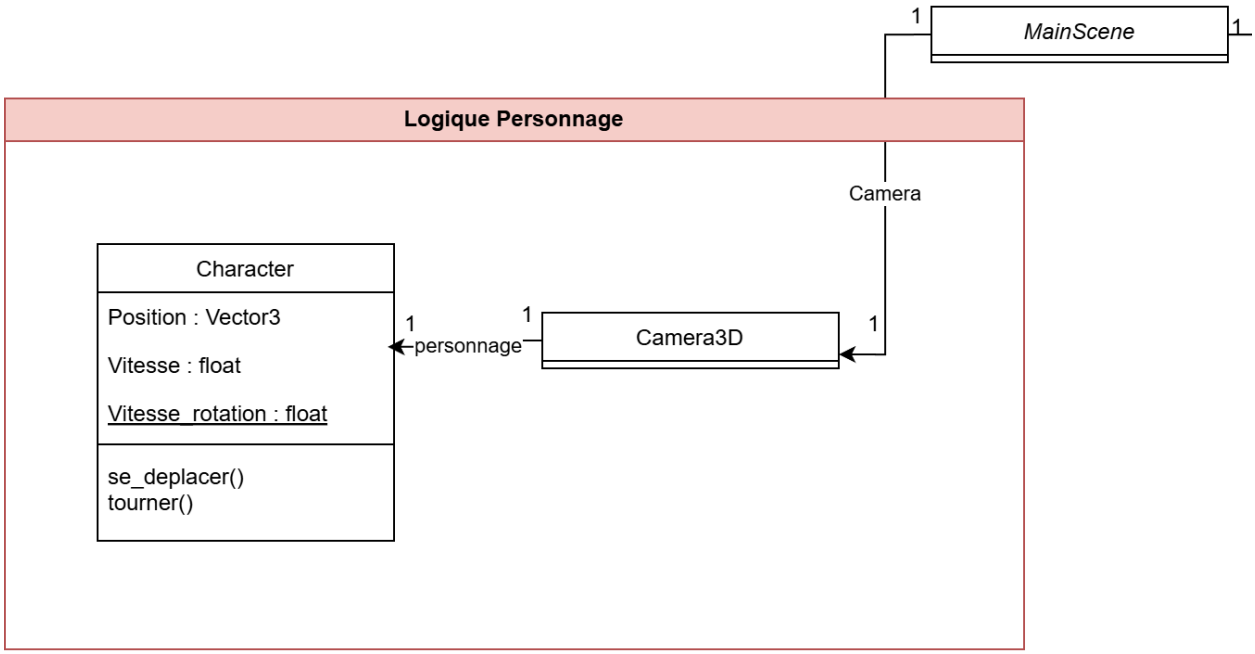
Diagramme de classes

Du fait que nous utilisons Godot, faire des diagrammes de classes nous est compliqué, Cependant l'architecture générale de l’application pourrait être la suivante :



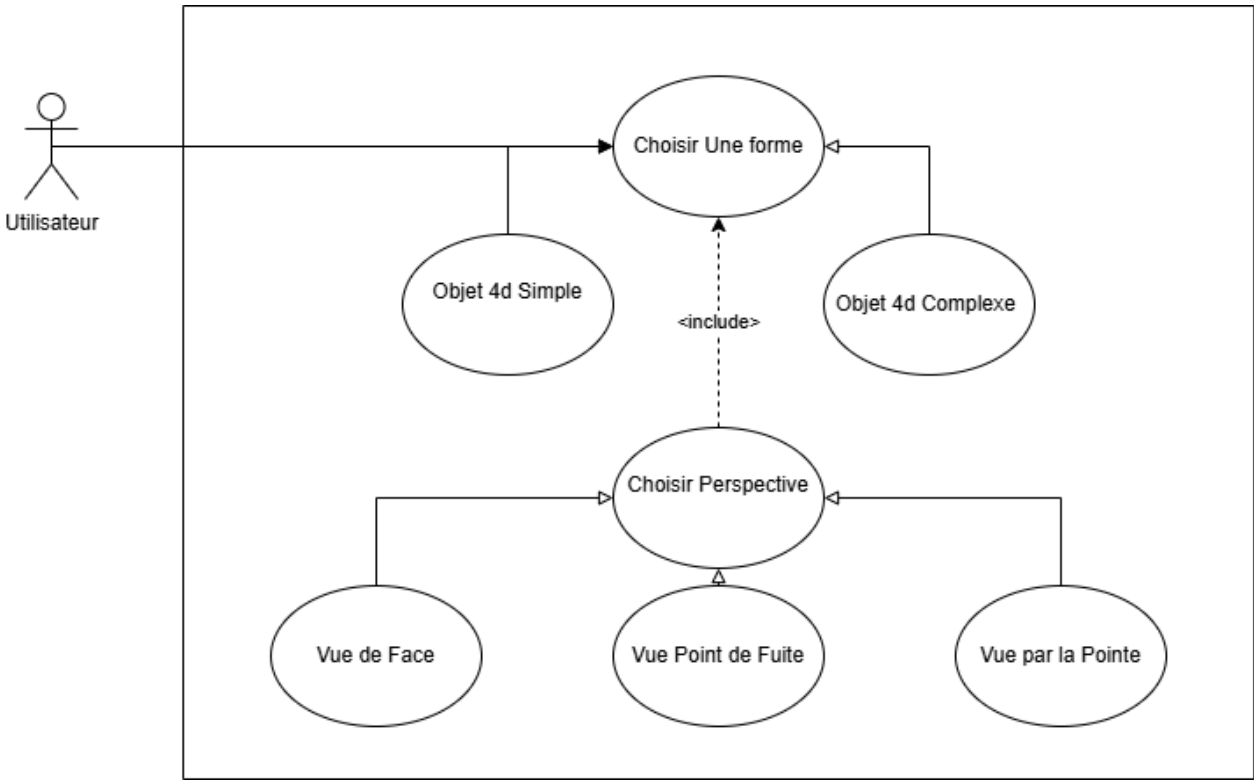

SousMonde
RigidBody3d : space
rotateFez()
addObject()
1
MainScene
1
sous-monde3d
*
1


Object4D
Object4D
translate()
rotate()
rotateFez()
display3D()
*
objects
Spherique4D
rayon : Float
Polyedre
verticles4D : Array<Vector4>
edge : Array<Edge>
Hypersphère
Hyperoctaèdre
Hypercube


1
MainScene
1
Logique Personnage
Camera
Character
Position : Vector3
Vitesse : float
Vitesse_rotation : float
se_deplacer()
tourner()
1
personnage
1
Camera3D
1

Cas d’utilisation

Cas D’utilisation Initialiser



Cas d'utilisation : Initialiser

Scénario principal :

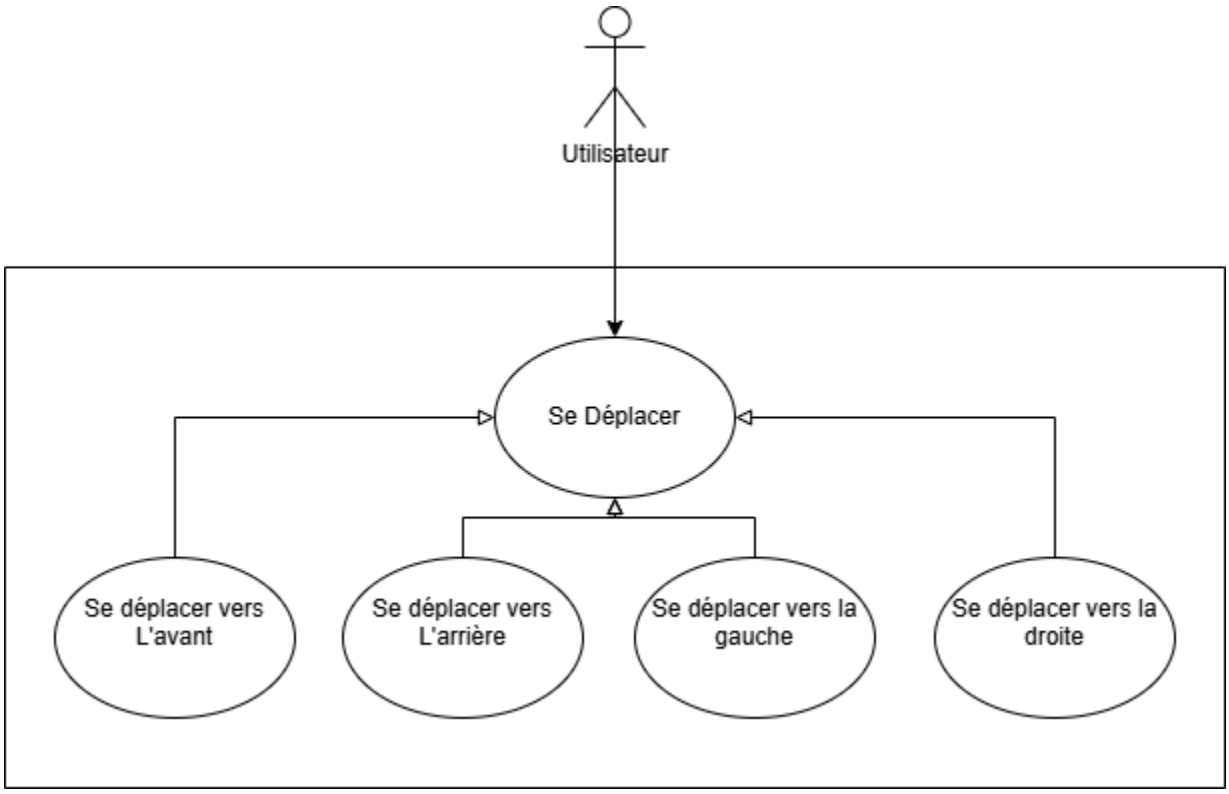
1. L'utilisateur démarre l'application et arrive sur l'écran principal.
2. L'application affiche une liste de formes géométriques disponibles (hypercube, hypersphère, etc.).
    - L'utilisateur sélectionne une forme parmi les options.
3. Une fois la forme choisie, l'application propose une liste de perspectives 4D possible
    - L'utilisateur sélectionne une perspective parmi les options.
4. L'utilisateur valide ses choix en cliquant sur Visualiser.
5. L'application charge les paramètres choisis et démarre la simulation avec la forme et la perspective sélectionnées.

Scénarios alternatifs :

- **Si aucune perspective n’est sélectionnée** :
  Une perspective par défaut est automatiquement assignée, et l'utilisateur est informé.

Conditions de validation :

- L'utilisateur peut naviguer entre les choix de forme et de perspective sans confusion.
- Le lancement de la simulation affiche correctement la forme et la perspective choisies.
- Les options non valides (aucune sélection) sont gérées par des messages explicites ou des choix par défaut.



Cas d'utilisation : Se déplacer

Scénario principal :

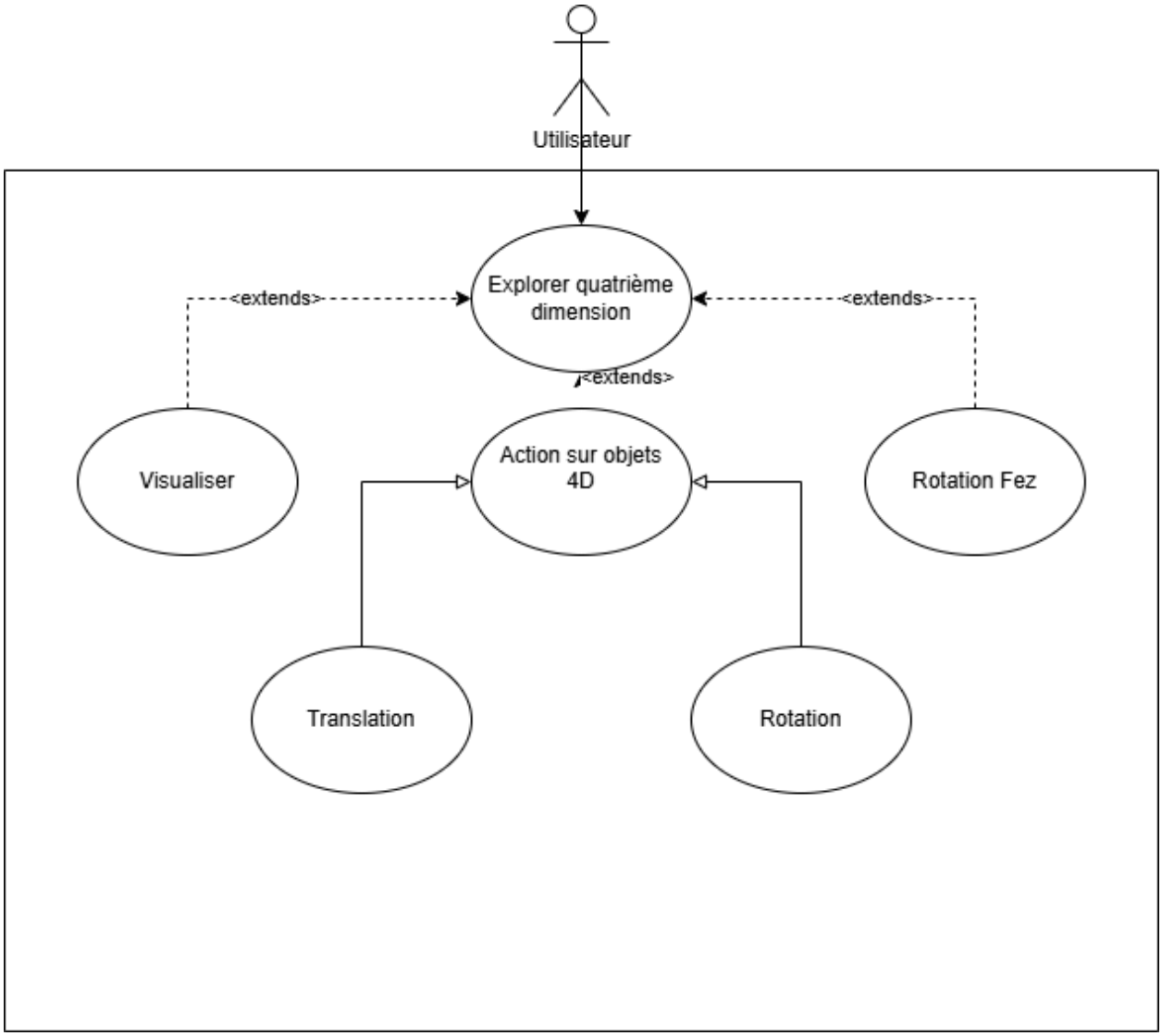
1. L'utilisateur commence dans un univers projeté en 3D.
2. Avec le joystick du contrôleur VR, il se déplace dans les trois dimensions classiques.
3. Il peut utiliser l’interface à la type Fez pour découvrir de nouvelles couches du monde

Conditions de validation :

- L'utilisateur peut naviguer dans les quatre dimensions sans confusion.
- Les transitions entre couches sont fluides et visuellement cohérentes.

- Aucun bug n'interrompt l'exploration.

Cas d'Utilisation Explorer des objets géométriques en 4D



Cas d'utilisation : Explorer des objets géométriques en 4D

Scénario principal :

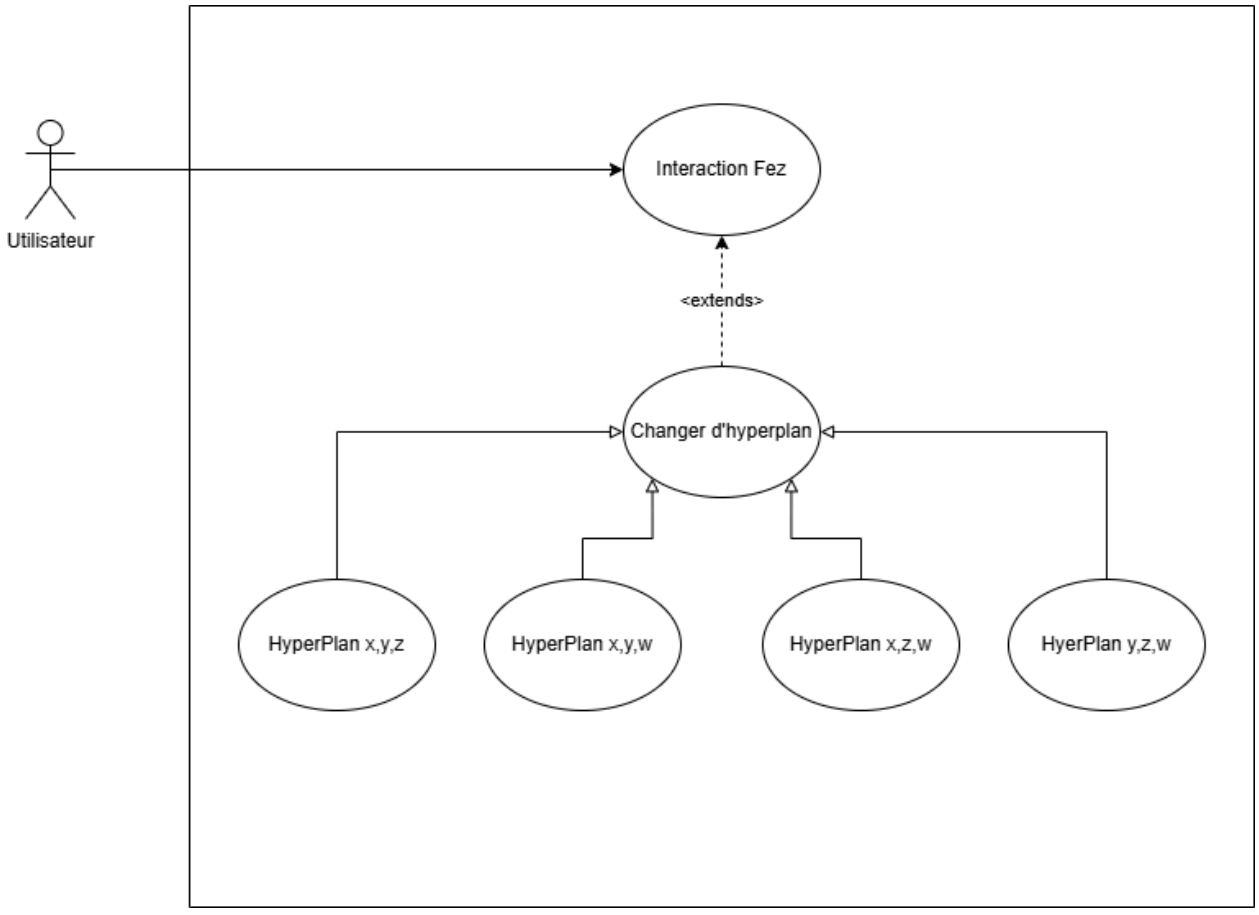
1. L'utilisateur démarre l'application et choisit une forme ainsi que la perspective souhaités

2. La forme apparaît projetée dans un espace 3D.
3. L'utilisateur utilise les contrôleurs VR pour :
    - Faire pivoter l'objet dans les trois dimensions visibles.
    - Déplacer l'objet

Conditions de validation :

- L'utilisateur peut voir l'objet sous différents angles et projections.
- Les contrôles sont intuitifs et réactifs.
- Les projections changent sans erreur graphique ni lag.

Cas d'utilisation Interaction Fez



Cas d'utilisation : Interaction Fez

Scénario principal :

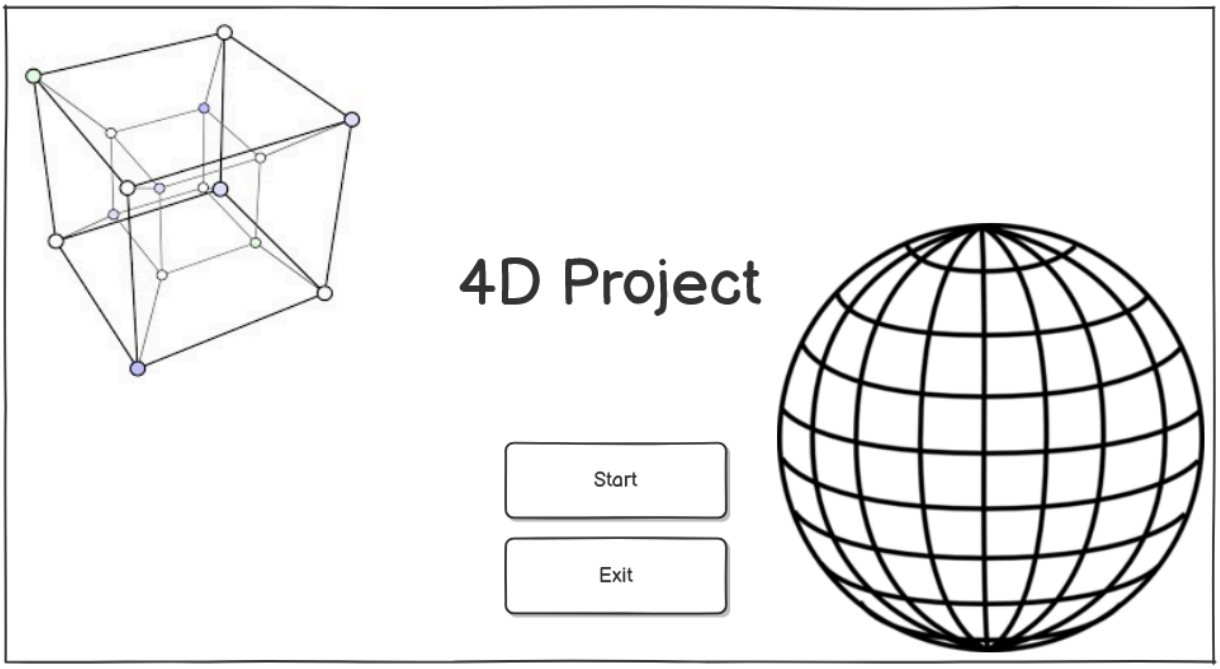
1. Le joueur contrôle un avatar dans un univers en 3D projetant une quatrième dimension.
2. En utilisant les commandes dédiées, il manipule les perspectives pour aligner des plateformes ou révéler des passages cachés.

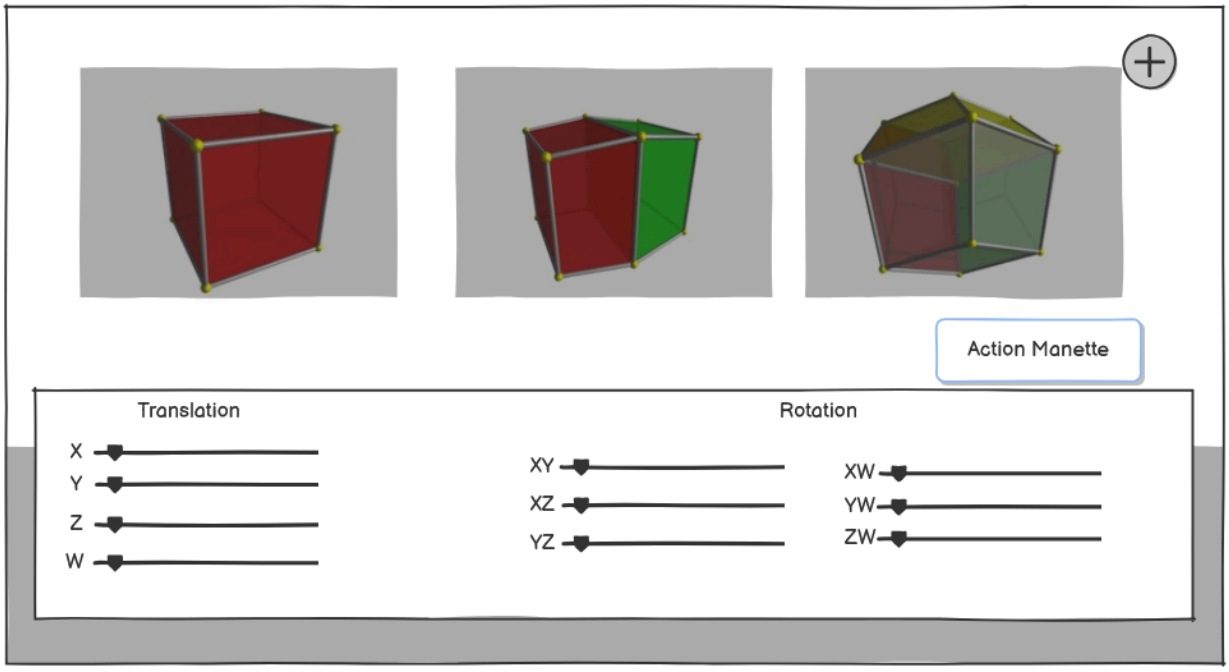
Conditions de validation :

- Les commandes pour interagir avec la 4D sont simples et intuitives.
- Les transitions entre perspectives sont fluides et immersives.

Maquette

Nous avons réalisé la maquette sur Balsamiq. Le lien public pour y accéder est disponible ici : https://balsamiq.cloud/sg1bjpy/pfjxzyz





Planning

Rôles

Dans le cadre de notre projet tutoré, nous avons décidé de désigner un chef de projet. Ce rôle est essentiel pour garantir une organisation structurée et la productivité de notre groupe. Le chef de projet est responsable des tâches suivantes :

- **Gestion des conflits** : Résoudre les désaccords au sein du groupe de manière constructive.
- **Prise de décisions** : Trancher sur les points stratégiques pour éviter les blocages.
- **Organisation du groupe** : Répartir les tâches et assurer une coordination efficace.
- **Planification des réunions** : Organiser régulièrement des réunions pour faire le bilan, ajuster les objectifs et s'assurer du bon déroulement du projet.
- **Coordination des activités** : S'assurer que toutes les parties du projet avancent de manière cohérente.

Nous avons également décidé que chaque membre du groupe participerait à toutes les étapes du projet, qu'il s'agisse de :

- **Programmation** : Développement des fonctionnalités.
- **Modélisation** : Création et manipulation des objets 4D.
- **Architecture** : Conception et organisation des différentes composantes techniques en lien avec l’utilisation de Godot.

Cette approche a été retenue pour plusieurs raisons :

- **Maintenir un niveau de compétence homogène** : Chaque membre travaille sur tous les aspects du projet, ce qui évite qu'un membre soit en retard ou perdu par rapport aux autres.
- **Faciliter la collaboration** : Une compréhension commune des différents concepts facilite le travail en équipe.

- **Adapter à la taille du projet** : Notre projet étant relativement petit et bien défini, il est plus efficace que chaque membre s'implique dans l'ensemble des tâches.

Chef de projet : Stanislas TROHA.

Développeur : Jawad KOMODZINSKI, Hugo RYSAK, Stanislas TROHA.

Documentaliste : Jawad KOMODZINSKI, Hugo RYSAK, Stanislas TROHA.

Itérations

Itération 1 (25/11/24 - 02/12/24) :

Temps total assigné à cette itération : 32 heures.

- Formation Godot. Création de jeux 2D et 3D (1 jeu 2D et 1 jeu 3D par personne).
- Compte-rendu sur ce qui a été appris, ce qui peut être utilisé pour mener à bien le projet.

Itération 2 (16/12/24 - 19/12/24) :

Heures totales assignées à cette itération : 26.

- Modélisation d'un hypercube
- Plusieurs perspectives.

Itération 3 (06/01/25 - 03/02/25) :

Temps total assigné à cette itération : 30 heures.

- Translation Hypercube
- Rotation Hypercube
- Interface utilisateur

Itération 4 (03/02/25 - 12/02/25) :

Temps total assigné à cette itération : 34 heures.

- Déplacement de la caméra
- Implémentation autre forme
- Début de l'implémentation de l'interface fez
- Implémentation des "sous-mondes" 3d

Itération 5 (24/02/25 - 24/03/25) :

Temps total assigné à cette itération : 20 heures.

- Interface Fez
- Début d'implémentation Vr

Itération 6 (24/03/25 - 31/03/25) :

Temps total assigné à cette itération : 26 heures.

- Mise en place de la Vr
- Polissage interface Fez

Itération 7 (31/03/25 - 06/04/25) :

Temps total assigné à cette itération : 26 heures.

- Polissage de l'application
- Ajout de fonctionnalités supplémentaires si le temps le permet
- Modélisation d'une maison pour l'interface Fez

Notre manque d'expérience concernant Godot et ce projet en général, nous empêche de planifier chaque itération avec précision. Nous suivrons ce planning méticuleusement, mais certaines étapes peuvent changer au cours des itérations si nous manquons par exemple de temps pour certaines itérations.