IUT Nancy Charlemagne
Université de Lorraine
2 ter boulevard Charlemagne
BP 55227
54052 Nancy Cedex

Département Informatique

Application VR : Déplacements en 4 dimensions.

Projet tutoré de 3ème année.

Rapport de fin de semestre.

Jawad KOMODZINSKI
Hugo RYSAK
Stanislas TROHA

Tuteur : Laurent DUPONT
Année universitaire : 2024-2025

Table des matières

Introduction

Pour commencer, nous préférons préciser que notre camarade Jawad KOMODZINSKI a été présent et impliqué dans ce projet durant les 3 premières itérations, après lesquelles il a quitté l'IUT. Par conséquent son nom apparaîtra dans ce rapport, à l'exception de la partie « *Présentation d'un élément orignal dont vous êtes fiers* » puisqu'il ne sera pas présent pour en parler.

Ce document a pour objectif de faire un bilan détaillé de l'avancement de notre projet à la fin de ces quatre premières itérations, marquant la conclusion de ce semestre. Il présente à la fois les choix techniques et fonctionnels réalisés, ainsi que les difficultés rencontrées et les solutions apportées.

Présentation du projet

Dans le cadre de notre projet tutoré en troisième année de BUT Informatique, nous avons entrepris le développement d'une application interactive permettant de visualiser et manipuler des objets en quatre dimensions (4D). Ce projet s'inscrit dans une démarche exploratoire, visant à rendre plus accessible la perception de la quatrième dimension à travers des outils numériques et une approche ludique.

L'objectif principal de cette application est de permettre à un utilisateur d'explorer et d'interagir avec des hypercubes (tesseracts) dans un environnement en trois dimensions. Comme il est impossible d'observer directement un objet en 4D, notre solution repose sur des techniques de projection et des mécanismes d'interaction permettant d'étudier ces objets sous différentes perspectives.

L'application est construite autour de plusieurs fonctionnalités clés :

- Visualisation d'hypercubes projetés en 3D, avec différentes méthodes de projection (stéréographique, perspective, orthogonale).
- Manipulation des objets à travers des transformations (translations et rotations sur un ou plusieurs plans).
- Interface utilisateur intuitive, permettant de modifier les paramètres d'affichage et d'interaction.
- Mécanique inspirée du jeu Fez, permettant d'effectuer des rotations dans des sous-espaces tridimensionnels.

Structuration du développement

À l'origine, le projet était prévu en deux grandes phases :

1. Première phase : Implémentation de la visualisation des hypercubes, application de transformations (translations, rotations), et développement d'une interface permettant de modifier la projection et les interactions.

2. Deuxième phase : Ajout de la navigation et du déplacement dans un espace 4D, en intégrant une mécanique de rotation similaire à Fez, ainsi que l'exploration d'une implémentation en réalité virtuelle (VR).

Toutefois, suite à notre étude préalable et aux premiers défis techniques rencontrés, nous avons ajusté cette organisation pour prioriser un produit fonctionnel à la fin du premier semestre. Nous avons donc concentré nos efforts sur :

- L'optimisation de la visualisation des hypercubes et leur comportement dans l'environnement 3D.
- L'amélioration de l'interface utilisateur pour garantir une interaction fluide et intuitive.
- L'intégration de la mécanique Fez en 3D, permettant d'expérimenter les rotations dimensionnelles sans encore introduire la VR.

L'aspect réalité virtuelle et certaines fonctionnalités avancées seront abordés dans la seconde partie du projet, en fonction du temps restant et des recommandations de notre tuteur.

Analyse

Découpage fonctionnel du projet

Voici les modules principaux qui composent notre application

- **Scène Principale** : C'est le monde où l'utilisateur va pouvoir se déplacer et interagir avec des objets.

- **Menu Principal** : Menu où l'utilisateur va pouvoir configurer la scène principale en choisissant les objets qu'il veut voir, leurs nombres, leur type de projection.

- **UI** : Dans le menu principal, l'interface utilisateur va permettre à l'utilisateur de sélectionner les objets qu'il veut voir. Dans la scène principale, l'utilisateur va pouvoir interagir avec les objets en utilisant l'UI par exemple en bougeant des sliders ou en appuyant sur des boutons pour modifier des valeurs (coordonnées, angle de rotation, …).

- **Sous-Monde** : Les sous-mondes permettent de délimiter un espace 3D. Un objet appartenant à ce sous monde disparaitra lorsque qu'il quittera les limites de ce dernier. On le définit comme une « fenêtre 3D » sur le monde en 4D qui contient la projection en 3D d'un objet 4D. C'est pour cela que si on le déplace trop selon une certaine coordonnée, il disparaîtra de la fenêtre de projection.

- **Hypercube** : Pour l'instant la seule forme 4D que nous avons implémentée. C'est une figure composée de 8 cubes connectés entre eux et de 6 faces carrées. On peut voir ça comme l'équivalent d'un cube mais à la quatrième dimension.

- **CharacterView** : La caméra qui représente en quelques sortes l'utilisateur. C'est en la déplaçant qu'on permet à l'utilisateur de se déplacer dans le monde afin d'explorer les différents hypercubes.

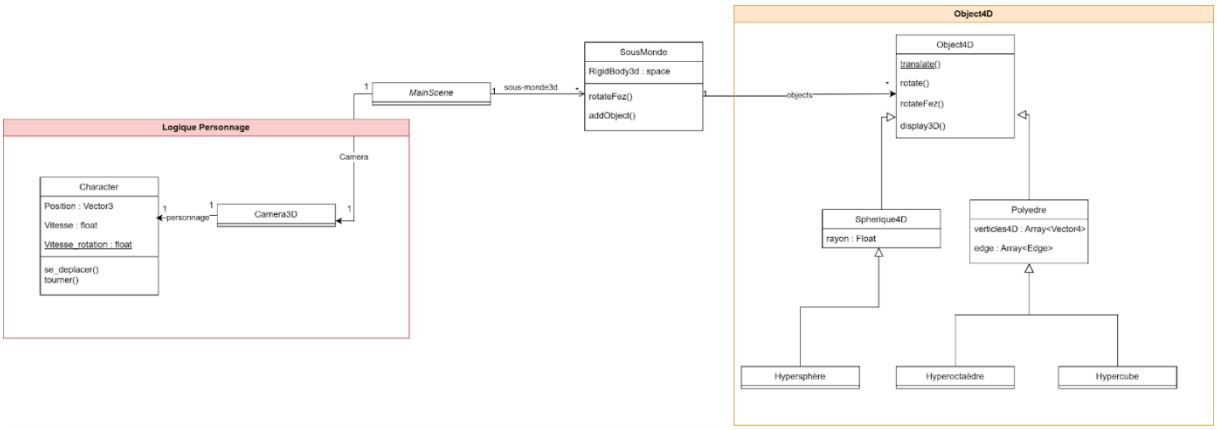
Technologies utilisées :

- **Godot** : Moteur de jeu

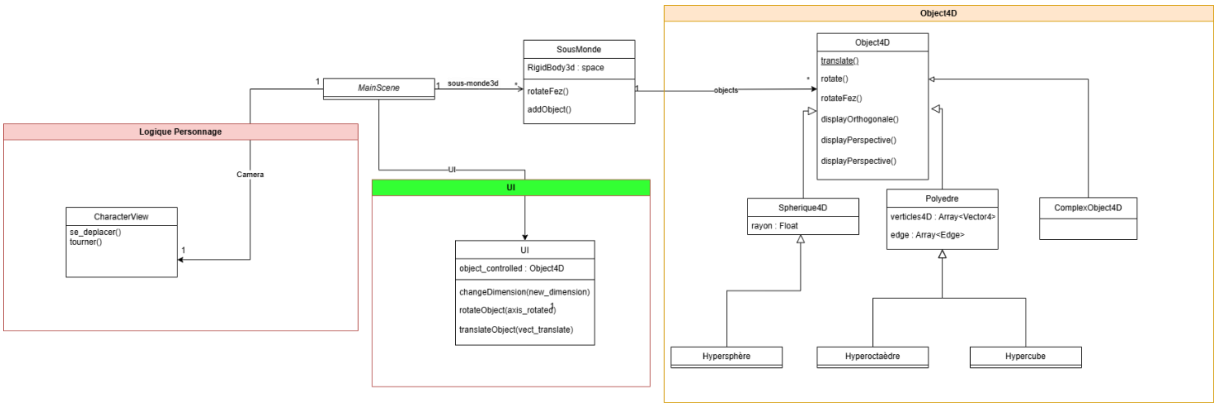
- **GDScript** : Langage de programmation

Diagrammes UML

Notre projet étant exploratoire, durant l'étude préalable nous avions seulement réalisé un diagramme UML pour représenter l'architecture de notre projet sur Godot :



L'étude préalable ayant été réalisée lorsque nous n'avions encore aucune base en Godot, avec le recul pris pendant ces 4 premières itérations nous aboutissons à ce genre de diagramme UML :



On remarque une certaine cohérence avec ce que nous avions fait au début, mais un manque d'information, comme l'interface utilisateur, ou des fonctionnalités représentées différemment dans notre code actuel par rapport à ce que nous nous étions imaginés.

Comme on peut le voir, nous avons choisi de réduire la logique personnage à seulement une caméra, puisque nous nous sommes rendu compte que pour la VR, il valait mieux contrôler un personnage à la 1ère personne qu'à la 3ème personne pour renforcer le sentiment d'immersion ainsi que la cohérence des commandes. A noter que nous avons rajouté la classe ComplexObject4D, elle représente en fait tous les objets 4D que nous souhaitons créer mais que ne sont pas polyèdres ou sphériques, ils hériteront alors directement de la classe Object4D. Et enfin nous avons rajouté la logique pour l'interface utilisateur.

Réalisation

Architecture logicielle

Notre projet a été développé en utilisant Godot. Godot est un moteur de jeu qui a la particularité d'être complétement open source, permettant de créer des jeux 2D et 3D. Il fut un moteur de choix pour notre projet car il est facile à comprendre et à utiliser, ce qui fut important puisque nous ne savions pas utiliser de moteur de jeu. Il a donc fallu apprendre à l'utiliser ainsi que son langage GDScript qui est aussi très facile à apprendre puisqu'il possède fortes ressemblances à Python. Godot fonctionne avec des scènes et des nœuds, ce qui facilite la conception modulaire, et a l'avantage de toujours être facile à appréhender.

Développement des fonctionnalités clés

Affichage d'objets 4D

Notre application nous permet actuellement de visualiser un seul type d'objet 4D, les « *tesseracts* », appelés également « *hypercubes* ».

Cet hypercube est un objet 4D qui peut être projeté en 3D via différentes techniques :

- **Projection orthogonale** : Si un point a comme coordonnée x y z w, on va le projeter sur l'espace 3D en fonction de ses coordonnées x y et z en oubliant le W. Cette projection garde les proportions mais n'a donc aucun effet de profondeur.
- **Projection perspective** : Consiste à projeter l'hypercube comme si on le regardait à travers une caméra placée dans un espace 4D. Il conserve l'effet de profondeur de l'hypercube mais ne garde pas les proportions. On peut le comparer à dessiner un cube sur une feuille, on projette un objet 3D sur un espace 2D.
- **Projection stéréographique** : C'est une autre manière de projeter un objet 4D en 3D. On projette les points de l'hypercube depuis un point spécifique sur un espace 3D. Cette projection permet de conserver les angles mais déforme fortement les distances à mesure que l'on s'éloigne du point de projection.

Vues

Il est possible de visualiser l'hypercube sous 3 vues qui sont les suivantes :

- **Vue de face** : C'est la vue où l'hypercube est aligné parallèlement aux axes 3D, En d'autres termes c'est la vue ou l'on ne peut voir qu'une des faces de l'hypercube parce que celle-ci éclipse toutes les autres. L'hypercube va donc apparaitre comme un cube à l'intérieur d'un autre cube relié par des arrêtes, cette vue permet une bonne compréhension de l 'hypercube
- **Vue Point de fuite** : Projection faite perpendiculairement à une face. Sous cette vue, le tesseract est formé de deux troncs de pyramide. Il est recommandé d'utiliser la projection perspective pour cette vue puisqu'on a besoin de la profondeur, cela marche cependant aussi avec une projection orthogonale.

- **Vue par la pointe** : On va orienter l'hypercube de manière que l'un de ses sommets soit orienté vers l'utilisateur. Toutes les autres parties vont être projetées autour de ce point. Avec cette vue, le tesseract va être formé de quatre volumes hexaédriques entourant le sommet.

Styles de l'hypercube

Nous avons aussi donné à l'utilisateur le pouvoir de choisir sous quel « style » il préférait visualiser l'hypercube. Nous en avons conçu trois :

- **Wireframe** : Ce style va représenter l'hypercube uniquement avec ses arêtes sous forme de lignes. Chaque sommet est connecté par des segments, sans face pleine. Cela donne une apparence ou toutes les connexions entre les points sont visibles.
- **Stylish** : Ce mode d'affichage est une version améliorée du style wireframe en ajoutant des sphères aux sommets et des cylindres pour les arêtes, il a l'avantage d'être plus visible et surtout plus agréable à regarder mais un peu plus gourmand en ressources. Lors des transformations (rotations, translations), c'est le style donnant une impression « mécanique » des transformations.
- **Solid** : Ici, l'hypercube est représenté en volume avec des face pleines, chaque face est colorée d'une différente couleur ce qui permet de mieux différencier les différentes facettes de l'hypercube. Contrairement aux modes stylish et wireframe, on ne peut voir l'intérieur de l'hypercube car aucune transparence n'est appliquée aux faces.

Transformations dans un univers 4D :

Concernant les transformations en 4D, bien que les objets soient projetés en 3D, les transformations qu'on peut leur appliquer concernent bel et bien un espace en quatre dimensions. Nous avons implémenté les transformations suivantes :

- **Translation** : L'utilisateur a la possibilité de déplacer l'hypercube selon 4 axes (x, y, z, w) ce qui lui permet d'explorer sa structure et d'observer comment l'objet évolue dans l'espace 4D.
- **Rotation** : L'utilisateur peut appliquer des rotations simples (autour d'un seul axe) ou double (autour de deux axes simultanément) à l'hypercube, lui permettant d'observer les transformations complexes de l'hypercube et d'appréhender sa structure sous différents angles de rotation.
- **Fez** : Inspiré du jeu Fez, ce mode permet à l'utilisateur de basculer entre différentes dimensions. L'utilisateur peut changer sélectionner une nouvelle dimension disponible pour basculer dans celle-ci et donc observer l'hypercube sous une autre dimensions 3D

Interface Utilisateur

- **Menu Principal** : Permet de choisir les objets à visualiser ainsi que leur méthode de projection et leurs styles d'affichage au démarrage de l'application.
- **UI dans le monde** : Permet d'interagir avec les objets 4D pour effectuer des transformations.

Tests et validation

Nous avons effectué plusieurs tests pour valider la projection correcte des objets 4D :

- Comparaison entre les différentes méthodes de projection : Nous avons vérifié la fidélité de chaque projection (orthogonale, perspective, stéréographique) en comparant les résultats obtenus à des représentations théoriques.
- Vérification de la cohérence des transformations appliquées : En effectuant des tests, nous avons vérifiés que les translations et rotation appliquées aux sommets de l'hypercube étaient cohérentes en termes de coordonnées et projections.
- Comparaisons avec un outil en ligne : Lors de la veille technologique nécessaire pour notre étude préalable, nous avions découvert un outil en ligne permettant d'effectuer des transformations sur un tesseract. Nous avons donc comparé nos résultats graphiquement à ceux obtenus avec cet outil en ligne.

Retours utilisateur

Pour tester l'interface utilisateur et vérifier qu'elle est facile et compréhensible pour tout néophyte, nous comptons faire remplir une fiche à plusieurs personnes pour obtenir des retours sur l'interface utilisateur et l'améliorer en fonction de ces derniers. Ces fiches seront disponibles lors des portes ouvertes où nous présenterons notre application aux visiteurs.

Difficultés rencontrées et solutions apportées

Compréhension de la 4D

L'un des plus grands défis de ce projet a été notre manque de connaissances sur la quatrième dimension. Contrairement aux dimensions 2D ou 3D, la quatrième dimension est complétement abstraite pour nous. En tant qu'humain, nous ne pouvons pas directement nous la représenter. Nous pouvons seulement nous représenter des projections de cette dernière. Il nous a donc fallu trouver les bonnes ressources pour appréhender ce concept, le comprendre et pouvoir avancer dans le projet.

Compréhension mathématique :

Manipuler des objets 4D, notamment pour les translations et rotations, requiert une compréhension avancée des mathématiques et de la géométrie, en particulier sur l'utilisation des matrices de transformation.

- Les matrices 4x4 sont essentielles pour représenter et appliquer des transformations linéaires dans l'espace 4D.
- La composition de rotations multiples implique des calculs complexes que nous avons dû tester et valider progressivement.
- Le fait de représenter la 4D en 3D était un grand défi pour nous. Au début, saisir les concepts des différentes projections (orthogonale, par perspective, stéréographique) était une étape essentielle.
- La représentation de ces données dans Godot, en GDScript, la façon dont les aborder était également une difficulté. Par exemple, comment représenter l'hypercube en termes de coordonnées ? Devons-nous stocker les faces ou les reconstruire à partir des sommets à chaque fois ?

Optimisation

Créer des objets 4D peut être couteux en performances, nous avons dû réserver une itération, la dernière de ce premier semestre, pour optimiser l'application du mieux possible, notamment pour les affichages d'objet en mode stylish qui consommait beaucoup de ressources et réduisait les performances de notre application. Dans cette même itération, nous avons également refactorisé du code, et fixé une multitude de bugs et warning relatifs aux manipulations des données dans notre code.

Planning de déroulement du premier semestre

Lors de ce premier semestre, nous avons dû reconsidérer les planifications réalisées lors de l'étude préalable tout en gardant dans les grandes lignes notre plan global.

1ère itération

Cette première itération, comme nous l'avions planifiée, était dédiée à notre apprentissage de Godot et son langage GDScript. Chaque membre du groupe a créé un jeu 2D pour se familiariser avec GDScript, les différentes structures de données, instructions possibles avec ce langage. Les jeux 2D ont également permis de découvrir le fonctionnement des nœuds de base, comment les lier entre eux, comment utiliser les signaux et évènements entre différentes entités. Pour poursuivre, chacun a ensuite créé un jeu en 3D pour intégrer une nouvelle notion de physique au développement du jeu, étant donné que nous étions assez familiers avec le moteur de jeu en lui-même. La gestion d'une nouvelle coordonnée a beaucoup impacté la physique, les graphismes et représentations d'objets dans le code, ce qui a été très utile pour la suite lorsqu'il était temps de commencer le développement du vrai projet.

2ème itération

Au moment de l'étude préalable et notre veille technologique et mathématique, nous avions pu découvrir les concepts clés de la 4D : Les transformations, projections, objets, etc.

Lors de cette deuxième itération, n'ayant pas encore assimilé précisément toutes ces notions, nous avons décidé de commencer la programmation simple d'un hypercube dans un espace 3D. Pour cela, il nous a fallu valider la compréhension des différentes projections de 4D en 3D. Le fait de pouvoir entamer la programmation nous a permis de nous rendre compte de la façon dont nous allions représenter un hypercube en termes de nœuds Godot, mais aussi en termes de script. Les Node4D n'existant pas, nous avons décidé de représenter un hypercube avec un Node3D, auquel on attacherait un script comportant les données réelles en 4D de l'hypercube.

3ème itération

Possédant une très bonne cohésion d'équipe et gestion de projet ainsi qu'une base de code suite à l'itération précédente, cette 3ème itération nous a permis de prendre une petite avance par rapport au planning que nous avions créé. Durant cette dernière, nous avons pu implémenter les translations, rotations simples et doubles, ainsi que l'interface utilisateur permettant non seulement d'effectuer ces transformations, mais aussi de générer les hypercubes de notre choix.

Cette itération a été décisive dans l'avancement de notre projet. C'est à partir de celle-ci que nous avions l'impression d'avoir une réelle application permettant de visualiser des tesseracts selon notre choix.

Pour un résumé condensé, voici toutes les fonctionnalités qui ont été réalisées :

- Transformations (rotations simples et doubles, translations)
- Interface utilisateur (menu principal, menu de transformation pour hypercube)

- Lien backend – UI (gestionnaire d'évènements, signaux…)
- Visualisation d'un hypercube sous différents styles (traits, cylindres & sphères, faces solides de couleur remplies).
- Sous-zones, pour donner cette impression de fenêtre sur le monde 4D depuis la 3D, cohérentes avec les transformations effectuées à l'hypercube.

4ème itération

Suite au départ de notre camarade, nous ne souhaitions pas réaliser beaucoup de nouvelle fonctionnalité lors de cette itération puisque nous avions compris qu'il fallait un produit fini et délivrable pour ce premier semestre. Ainsi, nous avons décidé de prioriser deux choses principales.

1. **L'interface Fez**, permettant d'effectuer des rotations sur des espaces pour visualiser différemment un objet 4D.
2. **L'optimisation**, **refactorisation** du code ainsi que le **fix** de certains **bugs**, **incohérences** dans le code qui auraient été traînés dans les itérations suivantes.

A noter que suite à une demande « ergonomique » et « pratique » de notre tuteur, nous avons également implémenté les points de vue pour cette itération. Pour rappel, ils consistent à pouvoir cliquer sur un bouton, dans l'interface utilisateur de transformation d'un hypercube, afin que le tesseract soit orienté selon le point de vue sélectionné (vue de face, vue de fuite, vue par la pointe).

Finalement, nous avons obtenu un produit fini, sans soucis de performance comparables à ceux rencontrés à la 3ème itération, et surtout sans bug. L'interface Fez fonctionnant parfaitement avec de petites animations. Nous pouvons remarquer que nous nous sommes plutôt tenus à notre étude préalable puisque toutes les fonctionnalités prévues pour ce premier semestre ont été réalisées, mis à part le fait de pouvoir visualiser d'autres objets que des tesseracts.

Avec une application optimisée et un code propre, nous sommes prêts à débuter cette deuxième phase de projet et remplir nos objectifs.

Répartition du travail entre étudiants

Nous avons, comme expliqué lors de l'étude préalable décidé de séparer les responsabilités en désignant un chef de projet qui s'engagerait à prendre en charge les tâches d'organisation et planification, bien évidemment en consultant les autres membres du groupe.

Afin de garantir une coordination efficace et de maximiser la synergie au sein de notre petite équipe, nous avons décidé d'adopter une approche collaborative pour l'ensemble du projet. Dès le départ, nous avons convenu que chaque membre devait contribuer à l'ensemble des composantes du projet — qu'il s'agisse de la programmation, de l'interface utilisateur ou de l'optimisation — afin que chacun développe une compréhension globale du système et puisse intervenir en cas de difficulté dans n'importe quelle partie du code. Les scripts s'entremêlant et les dépendances étant mêlées, il est essentiel que chaque membre du groupe saisisse tous les concepts de sorte à ce qu'il puisse intégrer sa solution/fonctionnalité sans perturber le reste du code.

Nom	**Rôle**	**Contributions clés sur les 4 itérations.**
KOMODZINSKI	Développeur	Déplacement de la caméra. Projection perspective. Rotations. Modes d'affichages.
RYSAK	Développeur	Projection orthogonale. Interface Utilisateur. Modes d'affichages. Interface Fez. Fix de bugs.
TROHA	Chef de projet / Développeur	Organisation et planification. Gestion des réunions et suivi du planning. Projection stéréographique. Translations. Lien backend UI. Modes d'affichages. Optimisation du code et fix de bugs.

Présentation d'un élément orignal

Interface Fez – Hugo RYSAK

Dans le cadre de notre projet une des fonctionnalités les plus innovantes que nous avions à réaliser et développer est l'interface Fez.

Une interface Inspirée du jeu Fez :

Le jeu Fez, développé par Polyton corporation et sorti en 2012, est un jeu de plateforme avec une mécanique bien particulière, bien que le monde où se trouve le personnage est en 3D, le joueur ne peut se déplacer que sur des plan 2D, d'ailleurs le joueur lui-même est un dessein en 2D. Un élément clé du jeu est la possibilité de faire pivoter l'univers autour de l'axe vertical, ce qui permet de révéler de nouveau chemin et de résoudre des énigmes. Ce jeu transforme donc un espace 3D en suite de projection 2D manipulable ce qui donne une illusion de profondeur et de connexions entre les perspectives.

Une interface inspirée de Fez pour la 4D

Dans notre projet de visualisation d'objet 4D, nous avons pu mettre en place une interface qui est inspirée de cette mécanique de jeu. L'utilisateur peut modifier la dimension affichée en sélectionnant des options dans un menu déroulant. Chaque sélection va donc changer la projection de l'objet 4D et changer la manière dont il est affiché, ce qui va permettre à l'utilisateur de pouvoir exploré cet objet et mieux en comprendre sa structure. Le fonctionnement de l'interface en détail est le suivant

1. L'utilisateur sélectionne une dimension dans la comboBox.
2. L'interface met à jour les dimensions disponibles en fonction de la nouvelle dimension.
3. L'hypercube est animé pour effectuer une transition fluide entre les deux projections.
4. Le maillage est reconstruit à chaque étape pour avoir une mise à jour réaliste et progressive.

Fonctionnement de l'interface :

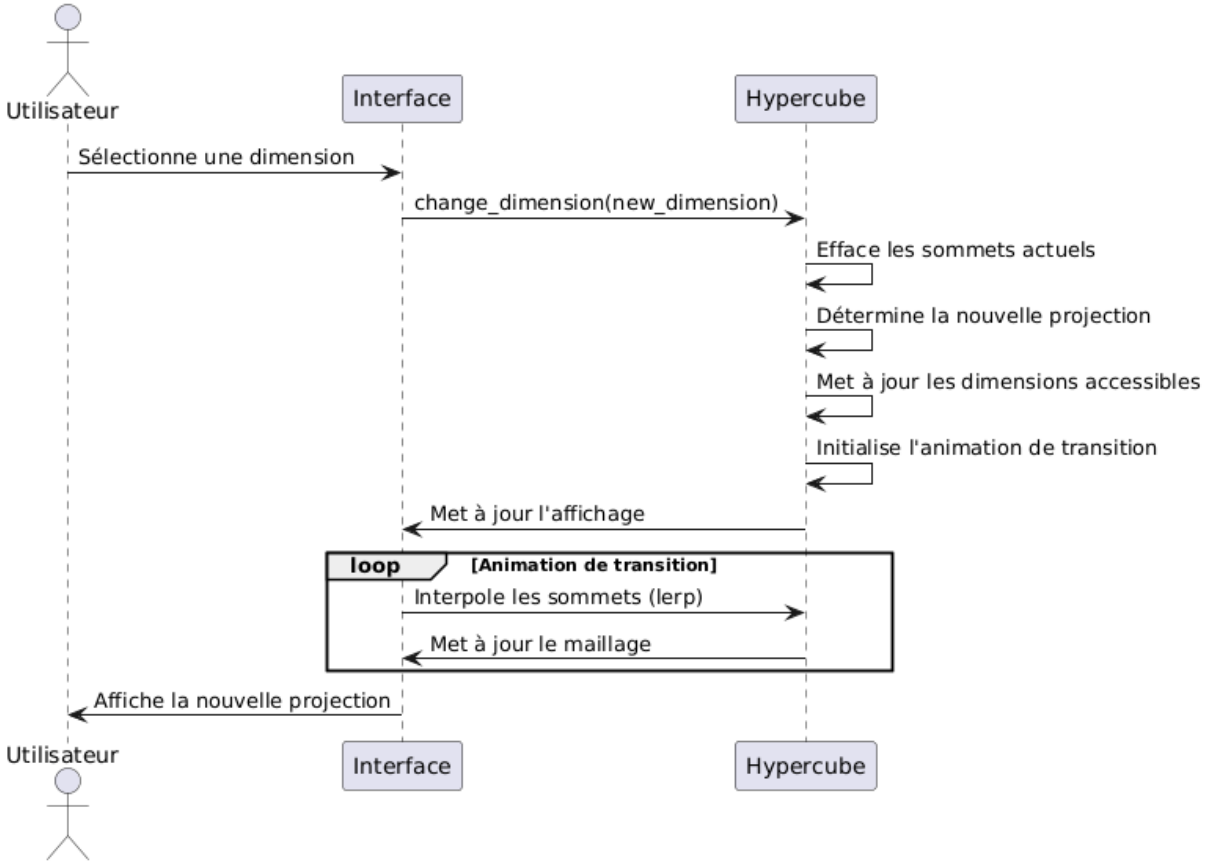
L'interface en jeu contient une comboBox listant les différentes dimensions de l'hypercube disponibles. Lorsque l'utilisateur sélectionne une nouvelle dimension, l'hypercube va donc être recalculé, et une animation fluide va animer la transformation de l'hypercube.

Gestion du changement de dimension dans l'hypercube :

L'hypercube possède une liste de dimensions possibles, qui sont définies par des permutations des axe x y z et w. Lorsque l'utilisateur va changer la dimension, le système va :

1. Effacer les sommets actuels.
2. Déterminer les nouvelles coordonnées des sommets en fonction de la nouvelle dimension.
3. Met à jour la liste des dimensions accessible en fonction de la nouvelle dimension.
4. Lance une animation de transition entre les anciens sommets et les nouveaux.

Voici un diagramme de séquence récapitulant le fonctionnement de l'interface Fez :

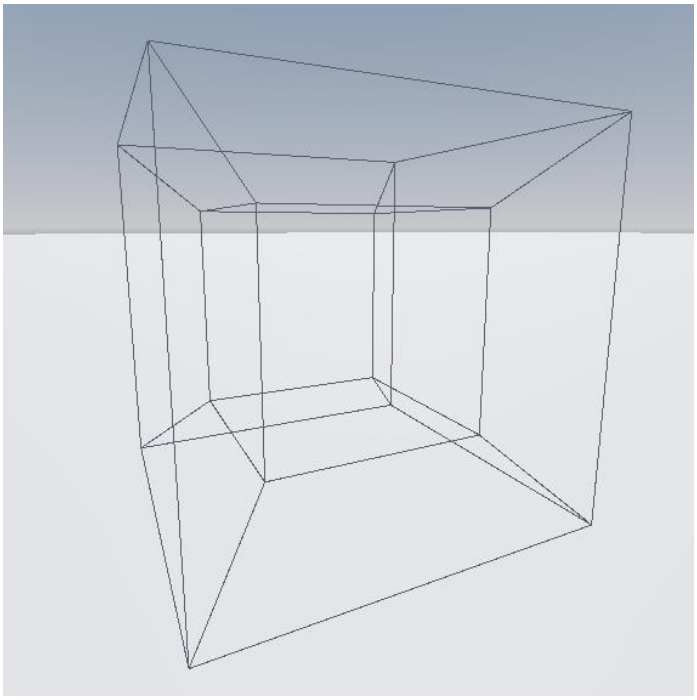


Styles d'affichage – Stanislas TROHA

L'un des aspects du projet dont je suis particulièrement fier est la création de plusieurs styles d'affichage pour les hypercubes. Cette fonctionnalité permet aux utilisateurs d'explorer ces objets en quatre dimensions sous différentes formes visuelles, chacune apportant un éclairage unique sur leur structure et leurs propriétés géométriques (avec ma préférence pour le mode stylish).

Nous avons implémenté trois **modes de rendu** distincts :

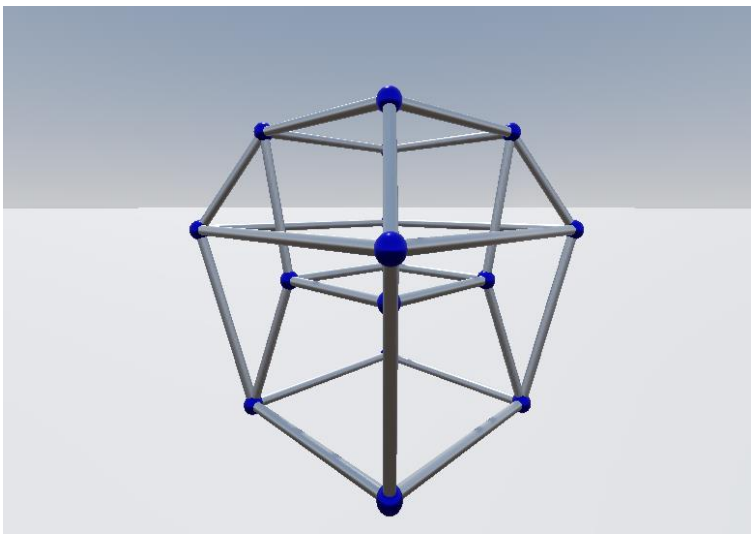
Mode Wireframe



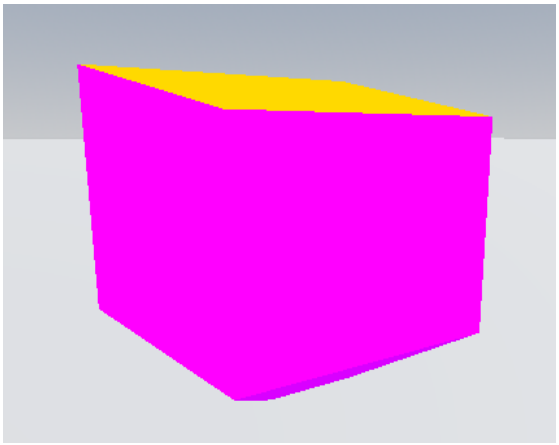
- Ce mode affiche uniquement les **arêtes de l'hypercube** sous forme de lignes.
- Il est particulièrement utile pour **analyser la structure interne** d'un hypercube, car il permet de voir toutes les connexions entre les sommets sans obstruction.
- Inspiré des **modèles classiques en mathématiques et en modélisation 3D**, ce type de rendu est souvent utilisé pour la **compréhension des polytopes** et pour visualiser la manière dont les dimensions s'interconnectent.

Mode Stylish

1. Une version améliorée du wireframe, où les **arêtes sont remplacées par des cylindres** et les **sommets par des sphères**.
2. Ce style donne une **représentation plus esthétique et immersive**, tout en conservant la clarté du wireframe.
3. Il permet de mieux **visualiser la profondeur** et les connexions entre les sommets, grâce à un effet plus volumétrique qui facilite la perception des relations spatiales.



Mode Full



- Ce mode affiche des **faces solides**, avec **des couleurs différentes pour chaque face**.

- Il est particulièrement utile pour **appréhender la géométrie globale de l'hypercube**, en mettant en avant ses différentes facettes tridimensionnelles.

- Ce mode se rapproche des **visualisations modernes en physique et en géométrie 4D**, où la colorisation aide à distinguer les différentes parties d'un objet multidimensionnel.

De plus, cette fonctionnalité a également été coûteuse en temps puisqu'au début de son implémentation, elle n'était pas optimisée et demandait beaucoup de ressources à l'application. Après de la refactorisation de code ainsi que de l'optimisation de structures de données et d'appels de la fonction de rafraîchissement *_process* plus aucun problème de performance n'a été détecté.

Cette diversité dans les styles d'affichage représente un élément clé de notre application, offrant à l'utilisateur une meilleure compréhension et immersion dans l'exploration de la quatrième dimension. Lors de portes ouvertes, je pense que certaines personnes posséderont des préférences différentes quant aux différents styles que l'on propose. C'est une avancée dont je suis particulièrement fier, car elle allie utilité pédagogique, flexibilité et qualité esthétique.

Fin du projet : Objectif à atteindre

Suite à ces quatre premières itérations, nous avons réussi à obtenir une application permettant de représenter des hypercubes sous différentes projections dans des sous-espaces. Pour rappel, on peut choisir la projection à utiliser pour chaque hypercube. Ensuite, on peut choisir comment le représenter avec trois styles différents (stylish, wireframe, full) et lui appliquer des translations, rotations (simples ou doubles) et des changements de perspective à la Fez. Le projet est sans bugs, optimisé et disponible à la démonstration.

Pour cette deuxième partie, soit les trois itérations suivantes, nous souhaitons prioritairement :

- Rendre l'application disponible en réalité virtuelle, pour une expérience plus immersive et ludique.
- Ajouter les empilements d'objets 4D pour construire des maisons par exemple, et autres formes et structures ludiques et intéressants à explorer.
- Polir l'application, améliorer l'interface utilisateur pour avoir un résultat satisfaisant et utilisable par n'importe quel néophyte de la 4D.
- Une optimisation finale.
- Ajout d'objets 4D supplémentaires (à choisir) pour construire de nouvelles structures.

Planning des itérations

Itération 5 (24/02/25 – 24/03/25)

Temps total assigné à cette itération : 20 heures.

- Implémentation VR de l'application
- Mise en place des formulaires pour la journée portes ouvertes du 1er Mars.
- Empilage de plusieurs objets 4D afin de créer des structures.

Itération 6 (24/03/25 – 31/03/25)

Temps total assigné à cette itération : 26 heures.

- Continuation de l'implémentation VR si ce n'est pas fini.
- Refactorisation potentielle de certaines parties code pour intégrer de nouvelles formes.
- Polissage de l'interface Fez.

Itération 7 (24/03/25 – 31/03/25)

Temps total assigné à cette itération : 26 heures.

- Polissage visuel final de l'application.
- Dernières optimisations des fonctionnalités ajoutées lors des itérations précédentes.
- Ajout de fonctionnalités supplémentaires si le temps le permet.

De par notre effectif réduit, nous avons reconsidéré les différents objectifs que nous nous étions fixés lors de l'étude préalable, et il est difficile de dire si nous réussirons à nous tenir précisément à ces estimations pour ces 3 prochaines itérations.

Conclusion

Au cours de ce premier semestre, nous avons réussi à poser les bases solides de notre application de visualisation et manipulation d'objets en 4D. Grâce aux différentes itérations, nous avons progressivement affiné notre compréhension des concepts liés à la quatrième dimension, en passant de la simple modélisation théorique à une implémentation fonctionnelle dans Godot.

Nos efforts ont permis de développer une application permettant d'afficher des hypercubes sous différentes projections, d'appliquer des transformations (translations, rotations) et de proposer une interface inspirée de Fez pour naviguer entre différentes perspectives. L'ajout de plusieurs styles d'affichage (wireframe, stylish, full) représente également une avancée importante pour une meilleure exploration de la 4D.

Malgré quelques difficultés rencontrées, notamment dans l'assimilation des concepts mathématiques et l'optimisation des rendus, nous avons su adapter notre approche pour obtenir un produit fonctionnel, stable et optimisé. Nous avons également revu certaines décisions techniques, notamment en simplifiant la gestion du personnage au profit d'une caméra libre, mieux adaptée à une future intégration VR.

Pour la suite du projet, notre priorité sera de transposer notre application en réalité virtuelle, afin d'offrir une expérience plus immersive et intuitive. Nous envisageons également d'ajouter de nouveaux objets 4D et d'améliorer l'interface utilisateur pour la rendre encore plus accessible.

Ce semestre nous a permis d'acquérir de nouvelles compétences en modélisation 4D, en programmation sous Godot et en gestion de projet. Nous sommes convaincus que les trois prochaines itérations nous permettront de finaliser notre application en respectant nos objectifs initiaux. Nous sommes fiers du travail accompli jusqu'ici et impatients de poursuivre ce projet dans la seconde phase de développement.