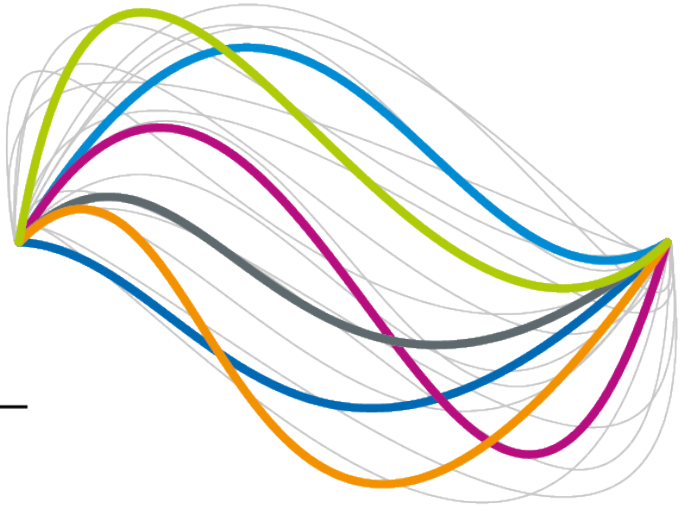


UBFC

UNIVERSITÉ
BOURGOGNE FRANCHE-COMTÉ

Compte rendu projet Synthèse d'image

(Poisson en 3D)

Membres du groupe :

AZROU Adlane

SAIRI Samir

Groupe : *TD1 (TP2)*

Prof encadrant : *Mme. Lanquetin S.*

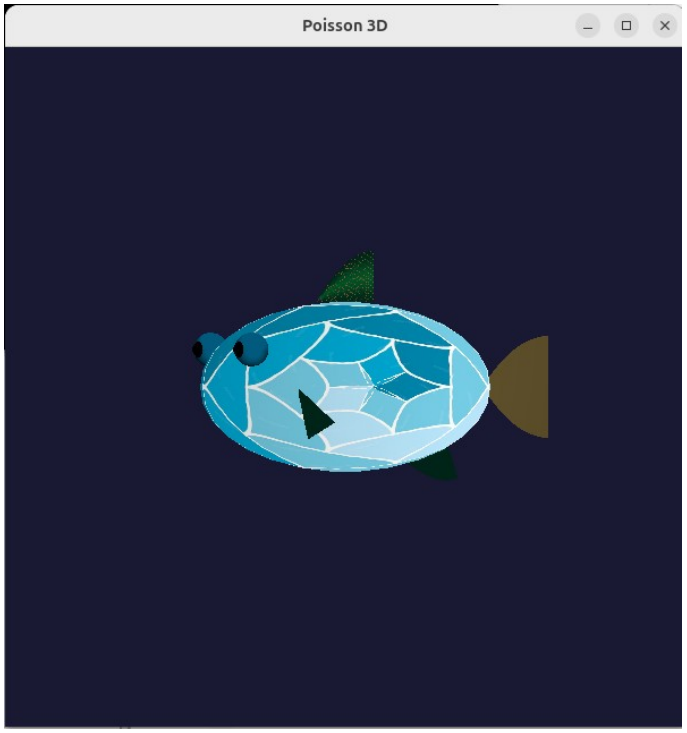
Année scolaire : *2024/2025*

Structure Générale du Programme

Le programme est organisé en plusieurs parties principales qui se complètent pour produire un rendu animé et interactif d'un poisson en 3D. Voici une vue d'ensemble des différentes composantes :

---

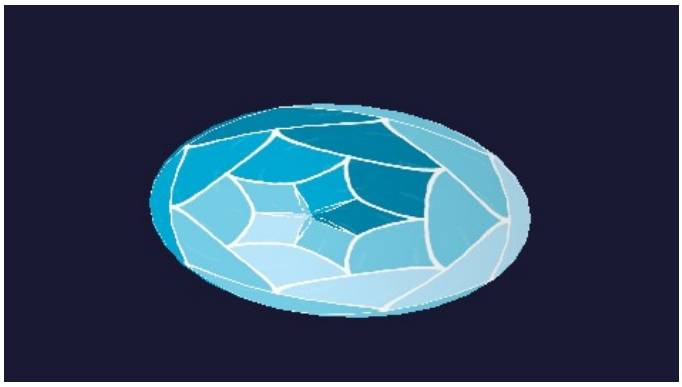
1. Modélisation du Poisson



Le poisson est modélisé en 3D en combinant des formes géométriques et des textures appliquées sur chaque partie. Chaque composant du poisson est construit de manière indépendante avant d'être assemblé dans une fonction centrale :

1,1. Corps du Poisson

- **Fonction** : `CorpsPoisson(double rX, double rY, double rZ)`
- **Description** :
    - Modèle central du poisson, représenté sous forme d'un ovoïde en 3D.
    - Paramétré avec des dimensions ajustables pour contrôler la longueur (`rX`), la largeur (`rY`), et la profondeur (`rZ`).



- **Construction** :

  Le corps du poisson est modélisé en 3D comme un ovoïde texturé, composé de quadrilatères (`GL_QUADS`) formant un maillage lisse. Voici les étapes principales de sa construction :

  - **Maillage** :
    - La surface est subdivisée en :
      - **Segments horizontaux (`numU`)** : Découpent la circonférence du poisson.
      - **Segments verticaux (`numV`)** : Découpent la hauteur du poisson.
    - Ces segments forment des quadrilatères qui recouvrent la surface.

  - **Calcul des Sommets** :
    - Les sommets sont calculés à l'aide de deux angles paramétriques :
      - `u` (horizontal) : Définit les segments autour de l'axe horizontal.
      - `v` (vertical) : Définit les segments le long de la hauteur.
    - Les coordonnées 3D des sommets sont obtenues avec :
      - $x = rX \cdot \cos(u) \cdot \sin(v)$
      - $y = rY \cdot \sin(u) \cdot \sin(v)$
      - $z = rZ \cdot \cos(v)$
    - `rX`, `rY`, `rZ` contrôlent la taille du poisson.

  - **Texture** :
    - Une texture (image JPEG) est appliquée pour représenter les écailles.

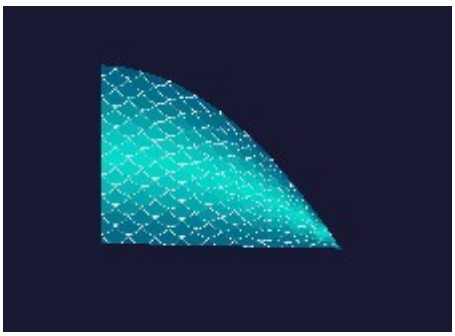
1.2. Queue du Poisson

- **Fonction** : `QueuePoisson(double longueur, double hauteur)`
- **Description** :
  - Forme losangique aplatie, attachée à l'arrière du corps, représentant la queue du poisson.
  - Oscille latéralement pour simuler le mouvement de propulsion.
- **Construction** :
  - Maillage quadrillé généré avec deux paramètres :
    - `longueur` : Contrôle l'extension de la queue en X.
    - `hauteur` : Contrôle la hauteur de la queue en Y.
- **Animation** :

- Mouvement de gauche à droite, contrôlé par la variable globale `angleQueue`, oscillant entre -30° et +30°.

1.3. Nageoire Dorsale

- **Fonction** : `NageoireDorsale(double longueur, double hauteur)`
- **Description** :
  - Nageoire incurvée située au sommet du poisson, au centre du corps.



- **Construction** :
  - Modélisée avec une bande triangulaire générée par un `GL_TRIANGLE_STRIP`.
  - Les sommets sont calculés pour simuler une courbure naturelle.
- **Texture** :
  - Texture spécifique (image JPEG) appliquée.

1.4. Nageoire Anale

- **Fonction** : `NageoireAnale(double longueur, double hauteur)`

- **Description** :
  - Située sous le corps du poisson, légèrement en arrière, cette nageoire contribue à l'équilibre du poisson.
- **Construction** :
  - Similaire à la nageoire dorsale, mais positionnée différemment.
  - Les sommets sont ajustés pour une courbure descendante naturelle.

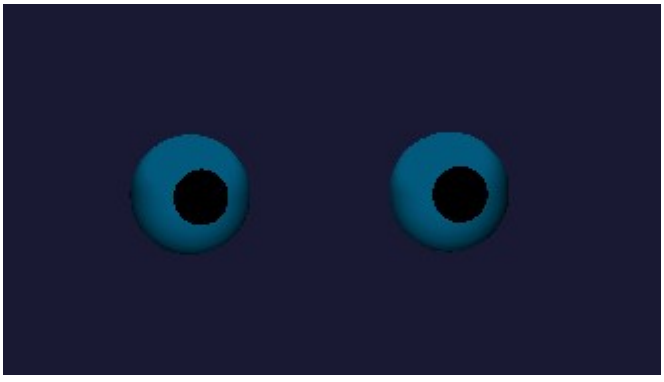
1.5. Nageoires Pectorales

- **Fonction** : `NageoiresPectorales(double longueur, double largeur, bool gauche)`

- **Description** :
    - Deux nageoires plates situées de part et d'autre du corps.
    - Ces nageoires oscillent de haut en bas pour simuler une propulsion réaliste.
- **Construction** :
    - Modélisées avec des triangles plats (`GL_TRIANGLES`), avec trois sommets :
        - Base (fixée au corps).
        - Sommet supérieur.
        - Sommet inférieur.
- **Animation** :
    - Le mouvement d'oscillation est contrôlé par la variable `angleNageoirePectorale`, oscillant entre -15° et +15°.

1.6. Yeux du Poisson

- **Fonction** : `YeuxPoisson()`
- **Description** :
    - Deux yeux sphériques placés sur les côtés avant du poisson.
    - Pupilles noires au centre des yeux.



- **Construction** :
    - Utilisation de `glutSolidSphere` pour modéliser les yeux et les pupilles :
        - Sphères bleues pour les globes oculaires.
        - Sphères plus petites et noires pour les pupilles.

1.7. Assemblage Global

- **Fonction** : `Poisson3D()`
- **Description** :
    - Intègre toutes les parties du poisson dans une seule structure.
    - Les différentes parties sont positionnées et orientées avec des transformations OpenGL (`glTranslatef`, `glRotatef`).
- **Étapes** :
    - **Corps** :
        - Dessiné au centre de la scène.
        - Texture appliquée.
    - **Queue** :
        - Attachée à l'arrière du corps.
        - Orientation dynamique en fonction de `angleQueue`.
    - **Nageoire Dorsale** :
        - Placée au sommet du corps.
    - **Nageoires Pectorales** :
        - Positionnées de chaque côté du corps.
        - Rotation dynamique en fonction de l'animation.
    - **Nageoire Anale** :
        - Positionnée sous le corps.
    - **Yeux** :
        - Placés symétriquement sur la tête.

---

2. Animation

Les animations du poisson sont gérées par plusieurs fonctions indépendantes qui simulent des mouvements naturels comme le balancement de la queue, les oscillations des nageoires, et le déplacement global. Ces animations sont ensuite synchronisées dans une fonction globale pour un rendu fluide et réaliste.

2.1. Animation de la Queue

Fonction : `AnimationPoisson()`

- **Description** :
    - La queue du poisson oscille latéralement entre -30° et +30°, imitant le mouvement de propulsion d'un poisson.
    - L'angle est contrôlé par la variable globale `angleQueue`.
    - La direction du mouvement est gérée par une variable booléenne `queueGauche`.
- **Mécanisme** :
    - Si la queue oscille vers la gauche, l'angle augmente de +2° à chaque mise à jour.
    - Lorsqu'elle atteint +30°, la direction change, et l'angle diminue jusqu'à -30°.

2.2. Animation des Nageoires Pectorales

Fonction : `AnimationNageoiresPectorales()`

- **Description** :
    - Les nageoires pectorales se déplacent de haut en bas entre -15° et +15°.
    - L'angle de chaque nageoire est contrôlé par la variable globale `angleNageoirePectorale`.
    - Le sens du mouvement est déterminé par une variable booléenne `nageoireVersHaut`.
- **Mécanisme** :
    - Si la nageoire monte, l'angle augmente de +2° à chaque mise à jour.
    - Lorsqu'elle atteint +15°, la direction change, et l'angle diminue jusqu'à -15°.

2.3. Déplacement Global

Fonction : `AnimationPoissonComplet()`

- **Description** :
    - Le poisson se déplace en oscillant sur l'axe Z pour simuler une nage sinusoïdale.
    - Il se déplace linéairement sur l'axe X, et effectue un **retournement** lorsqu'il atteint les limites (généralement à ±2.8 unités).
    - L'angle de rotation et la position sont contrôlés par les variables `positionX`, `positionZ`, et `angleRotation`.
- **Mécanisme** :
    - Si le poisson atteint les limites, il démarre un retournement en modifiant sa direction (`direction`).
    - Pendant le retournement, l'angle de rotation (`angleRotation`) change progressivement jusqu'à atteindre 180° (ou 0° selon la direction).
    - La position verticale (`positionZ`) suit une oscillation sinusoïdale pour simuler la nage.

2.4. Synchronisation Globale

Fonction : `AnimationGlobal()`

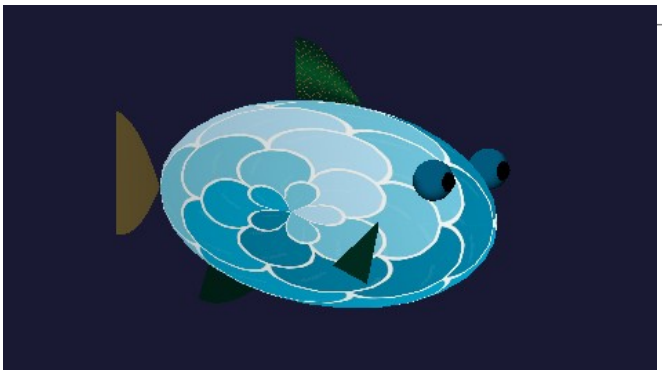
- **Description** :
    - Cette fonction centralise toutes les animations (queue, nageoires, déplacement) et agit comme une boucle de mise à jour.
    - Appelée via `glutIdleFunc`, elle garantit une coordination fluide entre tous les mouvements.Cette fonction est appelée dans main().

Toutes ces animations sont coordonnées dans une fonction d'animation globale qui agit comme une boucle de mise à jour.

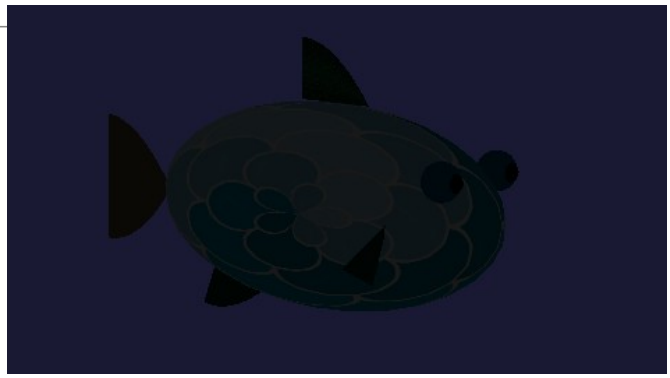
3. Éclairage

Le programme utilise un système d'éclairage basé sur **OpenGL** pour ajouter du réalisme au rendu du poisson en 3D. L'éclairage est configuré pour fournir une illumination naturelle en combinant une lumière ambiante douce et des lumières directionnelles positionnées autour du poisson.

Avec lumières



Sans lumières



3.1. Lumière Ambiante

- **Description** :
    - La lumière ambiante éclaire uniformément toute la scène, simulant une lumière environnementale douce.
    - Cela évite que des parties du poisson soient entièrement dans l’ombre.
- **Implémentation** :
    - La lumière ambiante globale est définie avec une faible intensité pour ne pas dominer les lumières directionnelles.

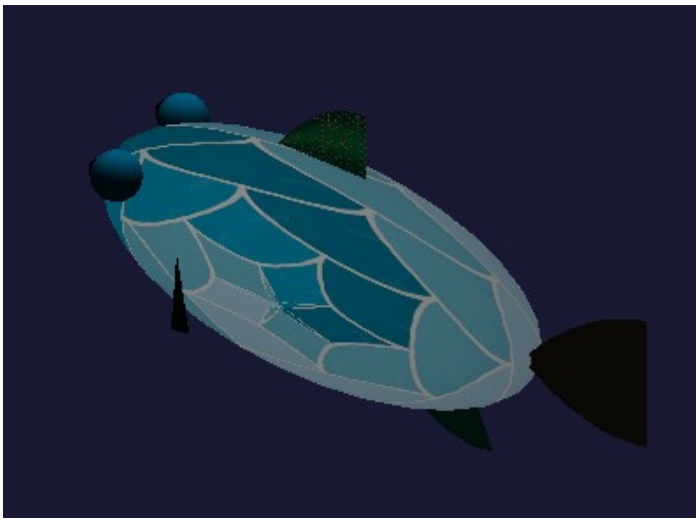
3.2. Lumières Directionnelles

Les lumières directionnelles ajoutent de la profondeur et des reflets au poisson. Chaque lumière est définie avec des composants **diffus**, **spéculaires**, et une **position** pour contrôler l'effet visuel.

3.2.1.Lumière au-dessus (GL_LIGHT0) :

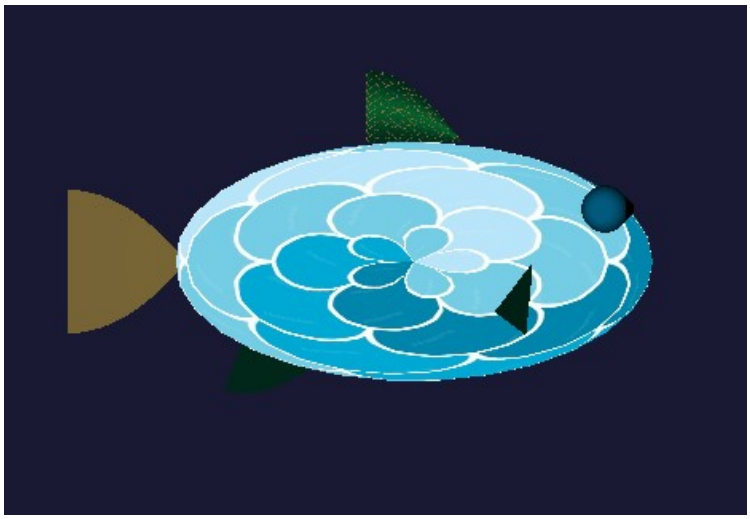
- **Position** : Placée au-dessus du poisson pour éclairer uniformément depuis le haut.
- **Caractéristiques** :
    - **Diffuse** : Une lumière blanche diffuse qui éclaire les surfaces directement exposées.

- **Spéculaire** : Ajoute des reflets blancs brillants pour simuler la réflexion sur une surface humide.
- **Position** : Positionnée au-dessus, à (0.0,5.0,0.75,1.0).



3.2.2.Lumières Latérales (GL_LIGHT1 & GL_LIGHT2) :

- **Position** :
  - **Lumière gauche (GL_LIGHT1)** : Positionnée à (0.0,0.0,−5.0,1.0).
  - **Lumière droite (GL_LIGHT2)** : Positionnée à (0.0,0.0,5.0,1.0).
- **Caractéristiques** :
  - Diffus et spéculaire similaires à GL_LIGHT0 pour un éclairage équilibré.
  - Ces lumières ajoutent des reflets latéraux, donnant du volume au poisson.



4. Textures

Les textures jouent un rôle essentiel dans l'amélioration du réalisme visuel du poisson, en ajoutant des détails tels que des motifs ou des écailles sur sa surface. Voici une description détaillée de leur gestion et application dans le programme.

4.1. Fichiers JPEG

- **Textures Utilisées** :
    - **Texture pour le corps** : Image représentant les écailles ou les détails du tronc du poisson.
    - **Texture pour la nageoire dorsale** : Image différente appliquée pour donner une apparence unique à la nageoire.
- **Fichiers** :
    - Ces textures sont chargées depuis des fichiers JPEG, par exemple :
        - `texture1.jpg` : Corps du poisson.
        - `texture2.jpg` : Nageoire dorsale.

4.2. Chargement des Textures

Le chargement des textures est géré avec la bibliothèque **LibJPEG**, qui permet de lire des fichiers JPEG et de les convertir en données utilisables par OpenGL.

- **Processus de Chargement** :
    1. **Fichier JPEG** :
        - Ouverture du fichier à l'aide de la structure `jpeg_decompress_struct`.
    2. **Décompression** :
        - Conversion de l’image en un tableau de pixels RGB.
    3. **Association à OpenGL** :
        - Création d’une texture OpenGL avec les données de l’image, puis elle est associée à l’aide des fonctions glGenTextures, glBindTextures, glTexParameteri, glTexImage2D.

4.3. Mapping des Textures

Une fois les textures chargées, elles sont mappées sur les surfaces 3D du poisson en utilisant des **coordonnées UV**.

- **Coordonnées UV** :
    - Chaque sommet du maillage 3D du poisson est associé à une position sur la texture (coordonnée UV).

    - Les coordonnées UV sont définies dans la plage [0, 1], où :
        - `U` correspond à l'axe horizontal de la texture.
        - `V` correspond à l'axe vertical de la texture.
- **Calcul des UV** : Les coordonnées UV sont calculées proportionnellement au maillage :
    - uTex=i/numU
    - vTex=j/numV, Ces valeurs sont ensuite passées à OpenGL via `glTexCoord2f`.

4.4. Application aux Composants

- **Corps du Poisson** :
    - La texture `texture1.jpg` est appliquée sur toute la surface ovoïde du corps.
    - Coordonnées UV calculées à chaque sommet.
- **Nageoire Dorsale** :
    - La texture `texture2.jpg` est appliquée uniquement à la nageoire dorsale.
    - La méthode de mapping est similaire à celle du corps.

---

5. Interaction Utilisateur

Le programme permet une interaction étendue avec l'utilisateur via des commandes **clavier**, **souris** et **touches spéciales**. Ces interactions modifient à la fois l'animation, le rendu visuel, et la caméra.

5.1. Commandes Clavier

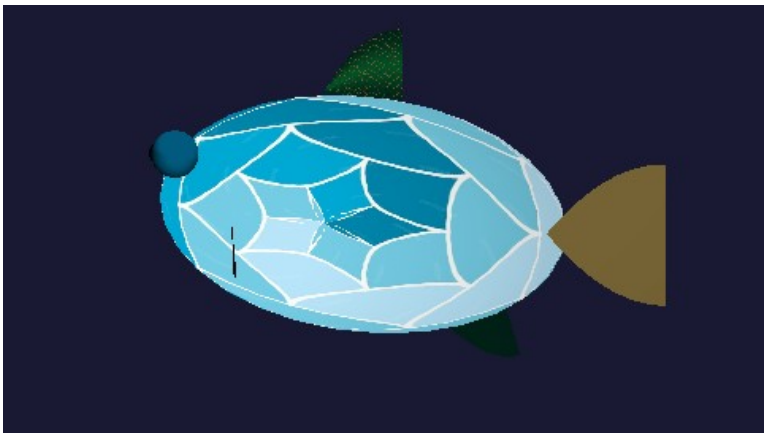
Les commandes clavier permettent à l'utilisateur de contrôler l'animation, le zoom, le mode de rendu, et les lumières. Voici une liste complète des actions possibles :

5.1.1. Contrôle des Animations

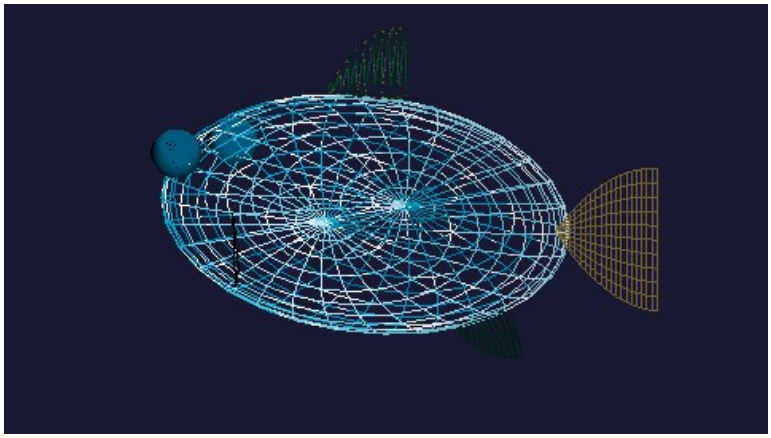
- **`b`** : Active l'animation de déplacement du poisson en mettant le boolean 'mouvementActif' à true, ce qui fait marcher la méthode AnimationPoissonComplet.
- **`n`** : Désactive l'animation de déplacement, mettant le boolean à false, ce qui bloque la méthode AnimationPoissonComplet.

5.1.2. Contrôle du Rendu

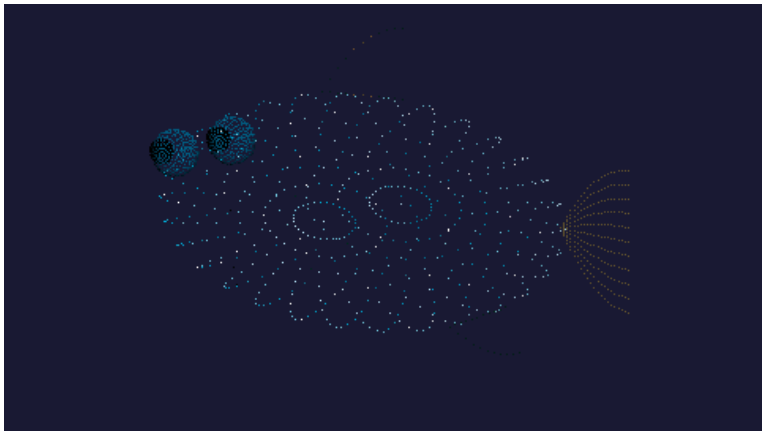
- **`p`** : Active le mode de rendu en **faces pleines** (mode par défaut).



- **`f`** : Active le mode **fil de fer** (wireframe), utile pour visualiser les structures géométriques.



- **`s`** : Active le mode **points**, affichant uniquement les sommets.



5.1.3. Contrôle du Zoom

- **`z`** : Effectue un zoom avant sur la scène.
- **`Z`** : Effectue un zoom arrière.

5.1.4. Gestion des Lumières

- **`h`** : Active ou désactive la **lumière du haut** (GL_LIGHT0), simulant une source lumineuse principale.
- **`g`** : Active ou désactive les **lumières latérales gauche et droite** (GL_LIGHT1 et GL_LIGHT2).

5.1.5. Quitter le Programme

- **`q`** : Quitte immédiatement le programme.

5.2. Commandes Souris

La souris est utilisée pour manipuler la caméra et explorer la scène 3D.

5.2.1. Rotation de la Scène

- En maintenant le **clic gauche** enfoncé :
  - Déplacer la souris vers la gauche ou la droite pour ajuster l'angle horizontal (`anglex`).
  - Déplacer la souris vers le haut ou le bas pour ajuster l'angle vertical (`angley`).

5.2.2. Clic et Glisser

- Lorsque le clic gauche est relâché, la rotation de la scène s'arrête.

5.3. Commandes avec Touches Spéciales

Les touches spéciales du clavier permettent de modifier l'orientation de la caméra.

- **Flèche Haut (`GLUT_KEY_UP`)** : Tourne la caméra vers le haut (modifie `angley`).
- **Flèche Bas (`GLUT_KEY_DOWN`)** : Tourne la caméra vers le bas.
- **Flèche Gauche (`GLUT_KEY_LEFT`)** : Tourne la caméra vers la gauche (modifie `anglex`).
- **Flèche Droite (`GLUT_KEY_RIGHT`)** : Tourne la caméra vers la droite.

---

6. Organisation des Données et Variables Globales

Les variables globales sont utilisées pour :

- **Position et Angles** :
  - Définir la position du poisson (`positionX`, `positionZ`) et les angles de caméra (`anglex`, `angley`, `anglez`).
- **État des Animations** :
  - Contrôler l'état actif/inactif des mouvements (`mouvementActif`).
- **Gestion des Lumières** :
  - Activer/désactiver les différentes sources lumineuses.
- **Textures** :
  - Stocker les dimensions et identifiants des textures pour le corps et les nageoires.

7.Conclusion

Ce programme offre une simulation graphique complète d’un poisson 3D, combinant modélisation précise, animations dynamiques et interactions utilisateur intuitives. Grâce à une structure modulaire et l'utilisation de textures et d'éclairages réalistes, il propose une expérience visuelle immersive. Il illustre efficacement les concepts clés de la programmation graphique avec **OpenGL**, tout en restant clair et extensible.