Matrices et transformations

Pour que le code du TP2 fonctionne, nous aurons besoin de certaines des fonctions implémentées lors du TP1. Si vous n'avez pas terminé le TP1, il est conseillé de le terminer avant de commencer le TP2.

Le code est ici

Pour importer du code écrit précédemment, vous pouvez utiliser `import` comme dans l'exemple ci-dessous. Notez que les deux fichiers (actuel et précédent) doivent être dans le même répertoire :

```
from exo1 import (
    initialize_camera,
    update_camera_position,
    dot_product,
   vector_length
)
```

Exercice 1

1. Importer les fonctions : cross_product, dot_product, vector_length et vector_normalize de TP1.
2. Assurez-vous que la fonction vector_normalize(axis) évite la division par zéro si un vecteur nul est fourni.
3. Trouver la fonction : scaling_matrix(axis, k) et implémenter la mise à l'échelle le long d'un axe arbitraire.
4. Trouver la fonction : rotation_matrix(axis, theta) et implémenter la rotation autour d'un axe arbitraire.
5. Trouver la fonction : orthographic_projection_matrix(axis) et implémenter la projection orthographique par rapport à un axe arbitraire.

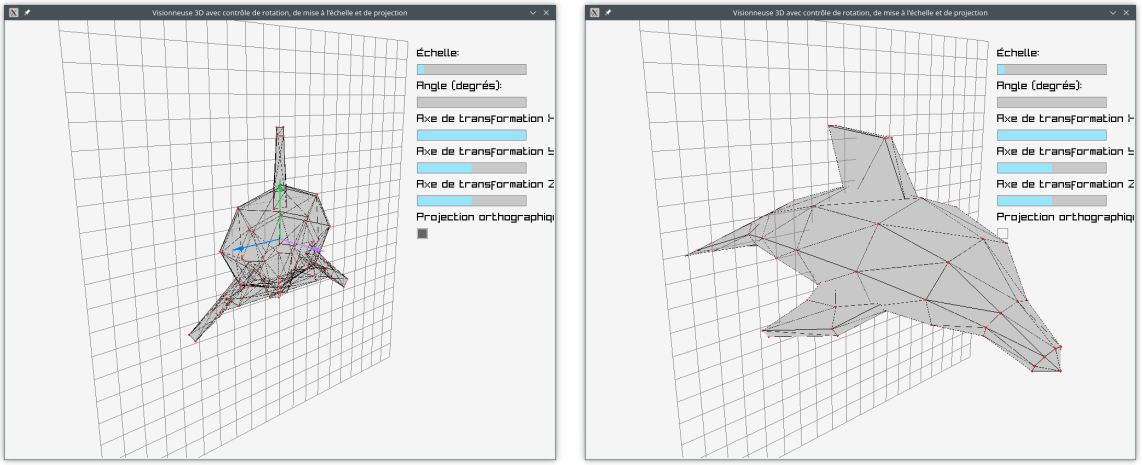

Fig. 1. Visualisation de la projection orthographique (à gauche) et de l'image inchangée (à droite).

Exercice 2

1. Trouver une fonction : apply_transformations(mesh, rotation_mat, scaling_mat, projection_mat) et expliquer les opérations. Comment le changement de l'ordre affectera-t-il les résultats ?
2. Pour la fonction de la question ci-dessus, expliquez la nécessité d'utiliser les sommets d'origine.
3. Trouver une fonction : draw_plane(axis, size=5, color=pr.GRAY) et expliquez la nécessité des deux produits vectoriels utilisés pour définir le plan de projection.

Exercice 3

1. Étendez le programme en ajoutant une transformation de cisaillement.
2. Expliquez et vérifiez pourquoi la mise à l'échelle arbitraire de l'axe n'a aucun effet sur la projection orthographique.