FASKA Rachid SAIDI Mokhliss IDU3

**Compte rendu TP3(séance 1)**

I/ PRÉPARATION AVANT TP

1. Utilisation de Pygame

Une image contenant texte

Description générée automatiquement(1)

La ligne 1 permet d’importer tous les modules pygame disponible. La ligne 2 permet d’initialiser chaque module importé. La fonction display.set\_mode() de la ligne 3 crée un nouvel objet qui représente les graphiques affichés. La ligne 4 réinitialise tous les modules pygame. On voit qu’une fenêtre se ferme en moins d’une seconde.

Une image contenant texte

Description générée automatiquement(2)

En exécutant ce programme, une fenêtre s’ouvre mais en appuyant sur un bouton quelconque du clavier, la fenêtre se ferme. La ligne 9 permet de « récupérer » tous les évènements. La ligne 11 fait passer la variable *continuer* à *false* si un évènement de type « appuyer sur une touche du clavier »(ligne 9) est réalisé. Donc tant que l’on n’a pas touché un bouton du clavier, la fenêtre restera ouverte.

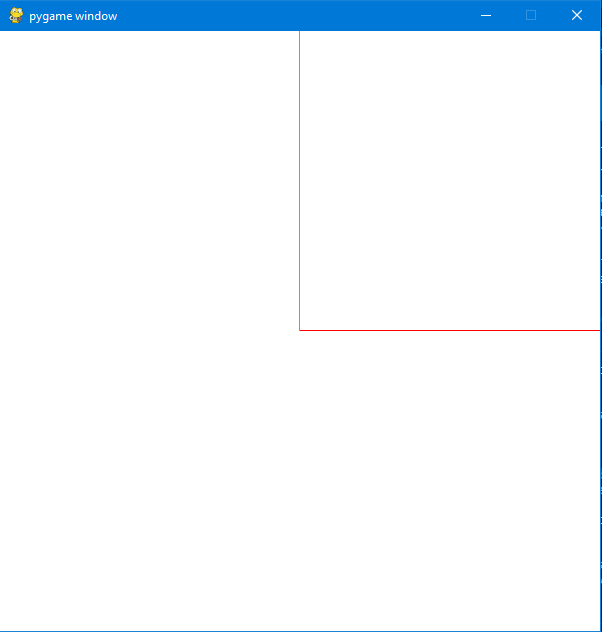
1. Utilisation de Pyopengl pour représenter des objets 3D

Une image contenant texte

Description générée automatiquement(1)

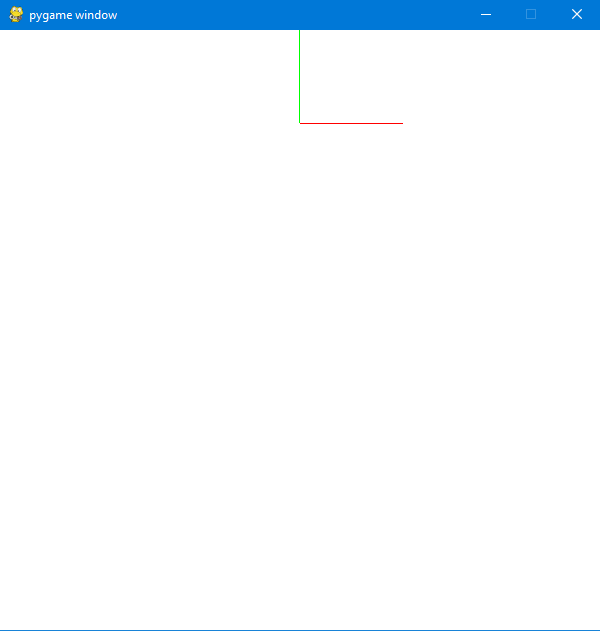
Une image contenant texte

Description générée automatiquement(2)

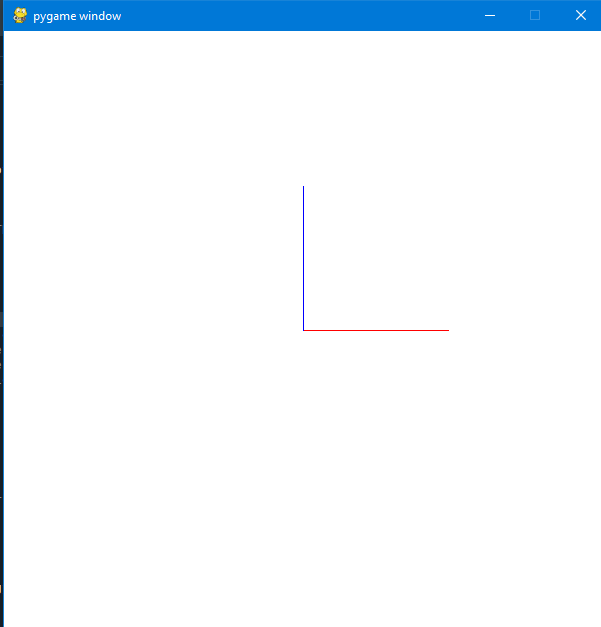
Résultat obtenu :

(3)

Résultat après translation :



Résultat après rotation :



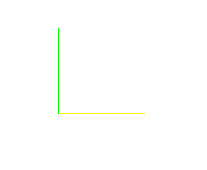
1. Découverte de l’environnement du travail du TP

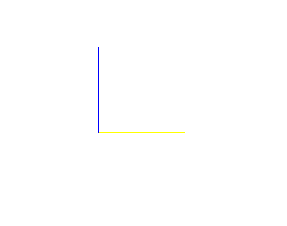
Une image contenant texte

Description générée automatiquement(1).c

On a rajouté la fonction *glRotatef* afin de pouvoir faire une rotation selon l’axe x.

Résultat obtenu :





II/ MISE EN PLACE DES INTÉRACTIONS AVEC L’UTILISATEUR AVEC PYGAME

Une image contenant texte

Description générée automatiquement1.d)

De la ligne 145 à 148 on permet le zoom ou le dezoom par les touches *PageUp* et *PageDown* du clavier.

1.e)

Une image contenant texte

Description générée automatiquement

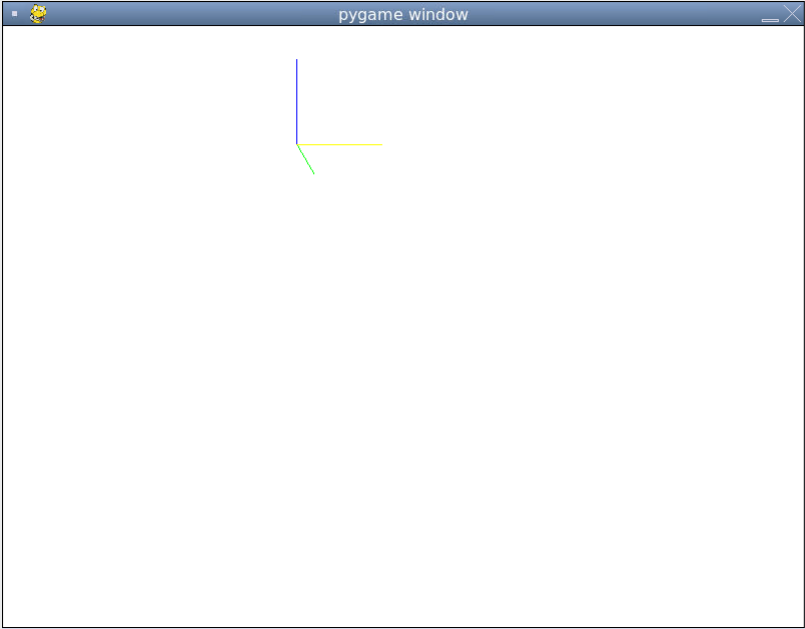
On veut obtenir un effet de zoom en utilisant la molette de la souris. Pour cela, on se sert de l’attribut *button* et du sens de l’action sur la molette associé à un évènement qui prend pour valeur 4 ou 5 selon le sens de l’action.

1.f)

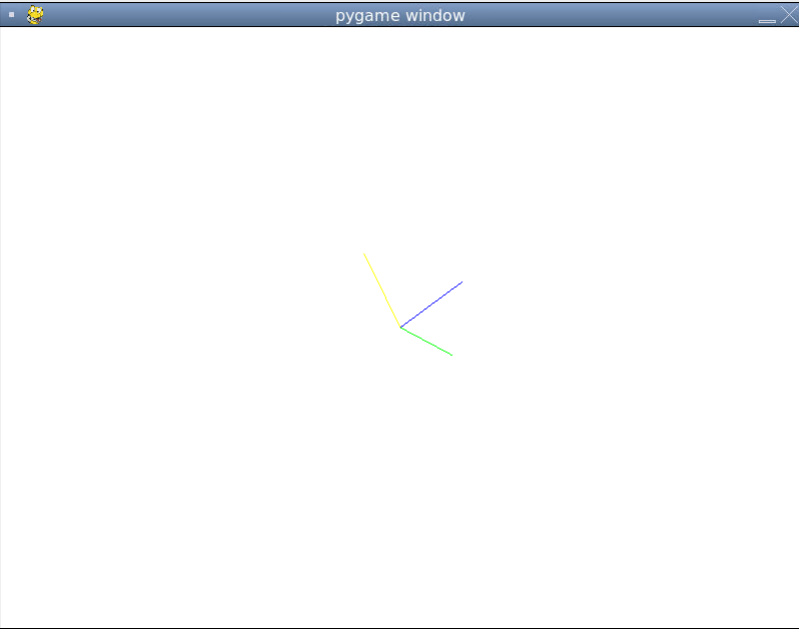
Une image contenant texte

Description générée automatiquement

La question 1)f a pour but de nous permettre de faire une rotation ou une translation quelconque par le déplacement de la souris et l’appui du bouton droit ou gauche. On utilise alors *pygame.mouse.get\_pressed()* pour savoir sur quel bouton sommes nous en train d’appuyer. On utilise *self.event.rel* pour permettre un déplacement quelconque selon l’axe x et z. Ensuite, selon le bouton sur lequel on appuie, soit on fait une translation avec *gl.glTranslatef* soit on fait une rotation avec *gl.glRotatef*. On divise par 25 pour ne pas avoir une translation « trop rapide ».

Résultat obtenu après une translation quelconque :

Résultat obtenu après une rotation quelconque :



III/ CRÉATION D’UNE SECTION

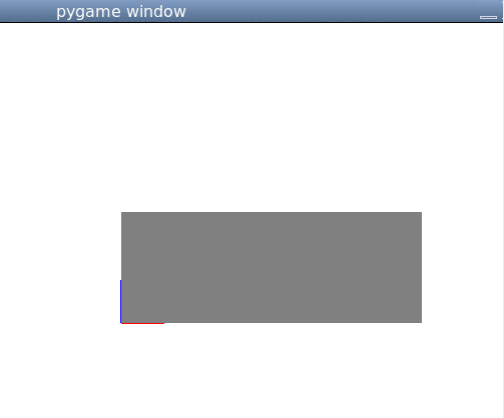
Une image contenant texte

Description générée automatiquement2.a)

2.b)

* Une image contenant texte

  Description générée automatiquementL’instruction ***Configuration().add(section).display()* permet de rajouter des paramètres de la classe *Section* à la classe *Configuration* et de créer un objet de type *Configuration* avec les paramètres ajoutés.**
* Le but de cette question est de pouvoir représenter une section. Pour cela, on réutilise le code donné en exemple mais on remplace les 1 par *self.parameters[‘width’]* ou par *self.parameters[‘height’]* selon la position du 1 donc *glVertex3fv* pour être en accord avec la fonction *Q2b()* du fichier *Main* où l’on a ajouté les paramètres *width* et *height*.

Résultat obtenu :

2.c)

Méthode *drawEdges*:

Une image contenant texte

Description générée automatiquement

C’est le même principe que la méthode précédente mais ici on utilise *gl.GL\_LINE* et non pas *gl.GL\_FILL* et on modifie la couleur (ici on a multiplié par 0.5).

Modification de *draw*:

Pour que la méthode ***drawEdges()*** soit exécutée en premier lorsque le paramètre ***edges*, il faut rajouter ces deux lignes au début de la méthode *draw()*:**



Résultat obtenu :

Des bords noirs sont apparus.