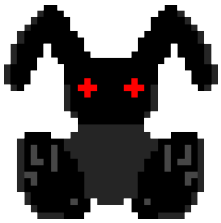


EPITECH.
L'ECOLE DE L'INNOVATION ET DE L'EXPERTISE INFORMATIQUE

Fil de Fer 2

Projection isométrique
Génération d'objets 3D

- Le Laboratoire aux Lapins Noirs -
lapinsnoirs@epitech.eu

Ce projet consiste à construire un programme affichant des objets en relief à l'écran. Les objets en questions sont générés par le programme.

Nom du dépôt de rendu : fildefer2

Index

1 – Détails administratifs

Votre dépôt de rendu doit s'appeler fildefer2.

Votre binaire doit s'appeller « fdf2 ».

Votre programme devra être compilé avec un Makefile.

Votre programme doit compiler avec -W -Wall -Werror.

Vous devez respecter la norme.

Votre rendu ne devra pas comporter votre programme compilé, de .o, #*# ou *~.

Les lignes de liaison avec les bibliothèques **doivent** être les suivantes :

```
-I/home/${USER}/.froot/include
-L/home/${USER}/.froot/lib
-llapin -lsfml-audio -lsfml-graphics -lsfml-window -lsfml-system -lstdc++ -ldl -lm
```

Votre programme **devra** s'exécuter en 1024*768 ou au-delà.

La quantité de RAM maximale autorisée **doit** être fixée dès le début du programme. Vous **devez** appeler bunny_set_max_ram en haut de votre main avec la quantité de RAM en octet demandé par le sujet. *Cette valeur est susceptible d'être modifié durant l'évaluation !*

Votre programme doit présenter un fonctionnement normal limité à **10Mo** de RAM soit 10485760 octets.

Vous **devez** appeler la fonction bunny_set_memory_check en lui passant true, cela avant de quitter votre programme ou encore plus tôt.

La taille **totale** de votre dépôt (Hors dossier .git) ne doit pas excéder **10Mo**.

Prenez garde à régler les droits d'accès de votre dépôt et de vos dossiers et fichiers.

2 – Fonctions autorisées

Ci-dessous, la liste des fonctions systèmes autorisées pour réaliser ce projet :

open, close, read, write
assert, alloca, setjmp, longjmp
srand, rand, time

Les libs math, dlfcn ainsi que le nécessaire à la mise en réseau.

Ainsi que les fonctions de la LibLapin présente dans lapin/basic.h

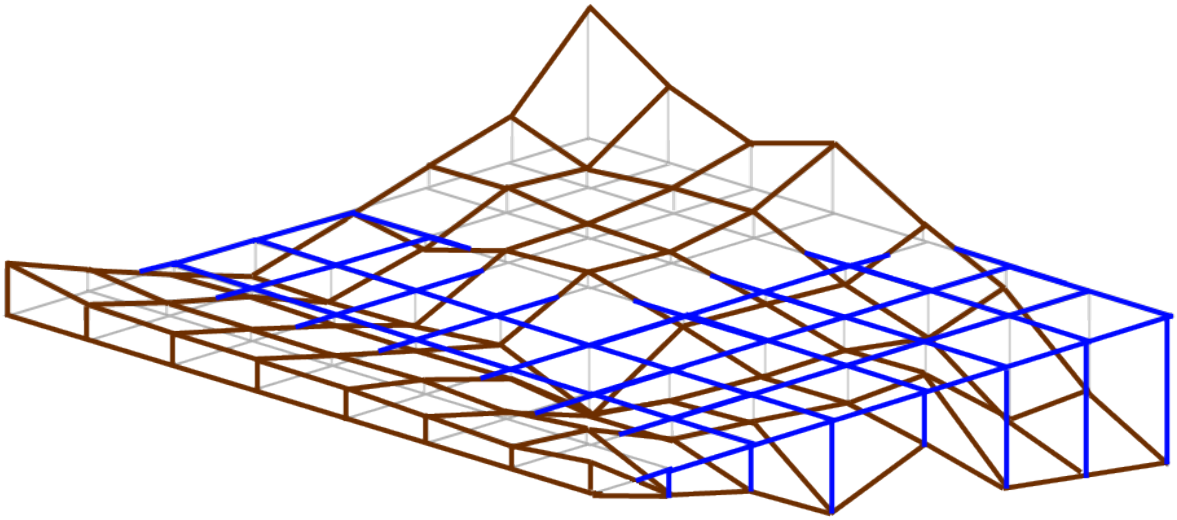
Toute fonction permettant d'allouer de la mémoire en dehors de bunny_*malloc, alloca ainsi que tous moyen de précipiter la fin du programme (exit...) sont **interdits**.

Les fonctions présentes dans lapin/advanced.h sont soumise à condition :

Toutes les fonctions **associées à un pointeur sur fonction** portant un nom type "gl_bunny_my*" doivent avoir été reprogrammée parfaitement. Une fonction disponible dans lapin/self_test.h vous aidera à vérifier leur fonctionnement. **Les autres sont autorisées.**

Si la fonction de test vous indique que votre fonction n'est pas une copie conforme, vous ne pouvez **pas** utiliser la fonction.

3 – Sujet



Le projet Fil de Fer 2 consiste à afficher à l'écran des objets en relief en utilisant une projection isométrique. Contrairement au Fil de Fer 1, plusieurs objets peuvent partager la même position. Les objets devront pouvoir être chargé depuis un fichier ou généré par le programme lui-même. Le format des fichiers pouvant être chargé est le suivant :

```
#technocore> cat -e file.ini
[forme1]$
type=fdf2$
shape=pixel$
data=$
0,0,0,$
8,9,1$
#technocore>
```

Les sections, comme ici « forme1 » contiennent les informations. Le champ type indique le format du fichier plus en détails. Le champ shape précise le type de forme à dessiner. Le champ data contient des coordonnées. Dans le cas d'un shape valant « pixel », les coordonnées sont par groupe de 3 : X, Y, Z. Dans le cas d'un shape valant « line », les coordonnées sont par groupe de 6. Dans le cas d'un shape valant « polygon », les coordonnées sont par groupe de 9.

Un champ color contenant les couleurs est susceptible d'être ajouté dans la section, faisant la même taille en terme de grappes de coordonnée (c'est à dire, le trio X, Y, Z) que data.

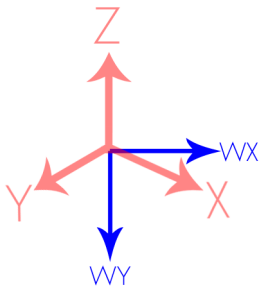
Plusieurs formes sont susceptible d'être contenu par un seul fichier.

Le chargement du fichier ne rapporte pas de points mais est obligatoire.

On doit pouvoir quitter le programme en appuyant sur la touche d'échappement.

3 – 1 Projection

La première étape consiste à afficher un terrain sans relief en utilisant une **projection 3D**. Une projection 3D est l'interprétation de coordonnées 3D (X, Y, Z) en coordonnée 2D (WX, WY) et donc représentable à l'écran.



Quelles sont les composantes de X, Y et Z dans le repère WX, WY ?
Essayez de dessiner le repère avec votre programme !

Tracer des lignes bêtement pour former une grille n'est pas une projection 3D.

3 – 2 Génération

La seconde étape consiste à générer des formes en 3D. Les formes à générer sont la sphère, le tore, le cône, le cylindre et le célèbre ruban de Möbius. Vous devrez également devoir permettre le dessin d'un parallélépipède.

Ces formes devront être modélise à l'aide de lignes. La modélisation sous forme de point vous rapportera tout de même la moitié des points.

3 – 3 Tri

La troisième étape est une étape de tri. Vous devez afficher les éléments à l'écran de manière à ce que les derniers dessinés soient les plus proches du point d'observation.

4 – Bonus

La liste des bonus ci-dessous n'est pas exhaustive :

Antialiasing

Arrondissement des angles.

Zoom.
Rotation (même partielle) autour de l'axe X.
Rotation (même partielle) autour de l'axe Z.

Édition à l'exécution de la position d'un point.

Translation des objets 3D.
Rotation des objets 3D.
Déformations des objets 3D.

Remplissage des formes avec de la couleur.
Dégradé de couleur sur les lignes entre deux points de couleurs différentes.
Dégradé de couleur sur les parallélogrammes entre quatre points de couleurs différentes.

5 – Interface de correction automatique

Afin de permettre une correction automatique de votre travail, nous vous demandons d'implémenter les fonctions suivantes :

```
void        tekpixel(t_bunny_pixelarray     *pix,
                     const t_bunny_position *pos,
                     const t_color          *color) ;

void        tekline(t_bunny_pixelarray      *pix,
                    const t_bunny_position  *pos,
                    const t_color           *color) ;

void        tekisoproject(t_bunny_position  *out,
                          int               x,
                          int               y,
                          int               z) ;
```

Ces fonctions peuvent tout à fait ne pas être utilisé dans votre projet si vous ne le souhaitez pas.

L'échec à la correction automatique vous fermera l'accès à l'évaluation.
Faites donc très attention en travaillant ces fonctions.

Ces fonctions doivent se trouver dans des fichiers .c situés dans le dossier ./tcore/, lui même situé à la racine de votre dépôt. L'intégralité du contenu de ce dossier et exclusivement de ce dossier sera compilé avec la moulinette.
Faites attention à ce qu'aucun main ne s'y trouve !

Si vos fichiers incluent des fichier en-têtes (.h), veillez à les placer soit à la racine de votre dépôt, soit dans un dossier ./include/, soit dans un dossier ./inc/ eux même à la racine.

tekpixel dépose dans pix un pixel de couleur color à la position indiqué dans pos.

tekline trace une ligne dans pix de pos[0] à pos[1] de couleur color[0] à color[1].
Si votre fonction ne gère pas le dégrade de couleur, utilisez seulement color[0].

tekisoproject écrit dans out les coordonnées projetées depuis x, y et z.

Le non-fonctionnement de l'une des trois fonctions est éliminatoire.