Porphyrographe : Nouvelles fonctionnalités pour l'été 2024

- un scénario complet *PG_full_scenario.csv* contient toutes les variables possibles utilisées pour contrôler la PG.
Tout scénario ne peut utiliser qu'un sous-ensemble de ces variables. Les variables supplémentaires ne seront pas prises en compte à moins qu'elles ne soient ajoutées au scénario complet et que PG ne soit recompilé après l'exécution du script *PG_cpp_source_generator*

- les données de l'ancien fichier d'en-tête ont été envoyées comme suit :

- les données du projet (client/serveur UDP, caméras, shaders) ont été ajoutées au scénario après les scènes et avant les données existantes.

- les variables du projet telles que la position de l'écran ont été ajoutées au scénario complet

- les variables de structure telles que la taille de l'écran ont été placées dans un nouveau fichier d'en-tête include
pg-header.h ainsi que d'autres constantes de *pg-all_include.h*

- le fichier *PG_source_generator.py* a été divisé en deux scripts :

- *PG_cpp_source_generator* génère du code C++ pour lier les variables C++ avec les variables glsl dans les shaders (utilisé pour recompiler PG après avoir ajouté de nouvelles variables dans le scénario complet).

```
# code généré à partir du scénario complet pour lier les variables C++ et GLSL
include/pg_script_header_Core.h (externes pour les variables de shader basées sur le scénario complet)
pg_render_body_Core.cpp (charge les valeurs des variables C++ dans les tables transmises aux shaders en tant qu'uniformes)
pg_script_body_Core.cpp (variables complètes basées sur le scénario et callBacks)
pg_shader_body_bind_Core.cpp (alloue des tables pour passer des valeurs de variables C++ aux shaders)
```

- *PG_glsl_source_generator* produit une nouvelle version des scènes dans laquelle les variables sont classées dans le même ordre que dans les scénarios complets.

- *PG_glsl_source_generator* génère le code des shaders glsl (versions complètes) pour lier les variables C++ aux variables glsl dans les shaders (utilisé pour générer de nouveaux shaders complets lorsque les codes des shaders sont modifiés).

Il n'est pas nécessaire d'utiliser ces générateurs avant d'exécuter porphyrograph si le scénario complet et les shaders ne sont pas modifiés.

- Les modules C++ ont été réorganisés en fonction de fonctionnalités plus fines, toutes les variables et fonctions globales ont été renommées avec le préfixe pg_. Les énumérations ont été préfixées par pg_enum_. Les modules sont

```
# global includes
include/pg-all_include.h
# traitement audio (PureData, PortAudio)
include/pg-audio.h
pg-audio.cpp
# fonctions génériques de rappel
include/pg-callBack.h
pg-callBack.cpp
# SVG clipArt (NVIDIA NV_path_rendering)
include/pg-clipArt.h
pg-clipArt.cpp
# gestion de la couleur et modulation de la couleur basée sur l'audio
include/pg-color.h
```

```
pg-color.cpp
# flashes entre les couches
include/pg-flash.h
pg-flash.cpp
# Gestion de l'interface graphique
include/pg-gui.h
pg-gui.cpp
# constantes
include/pg-header.h
# Initialisation des variables
include/pg-init.h
pg-init.cpp
# gestion de la lumière
include/pg-light.h
pg-light.cpp
# fonction principale et rappels GLUT
include/pg-main.h
pg-main.cpp
# gestion de maillages (réalité augmentée ou rendu 3D)
include/pg-mesh.h
pg-mesh.cpp
# messages affichés
include/pg-messages.h
pg-messages.cpp
# streaming de films et de caméras
include/pg-movieCamera.h
pg-movieCamera.cpp
# chemins enregistrement/lecture/sauvegarde
include/pg-path.h
pg-path.cpp
# image fixe (diaporamas ou clips animés)
include/pg-photoClip.h
pg-photoClip.cpp
# rendu GPU multipass
include/pg-render.h
pg-render.cpp
# gestion dynamique des variables par le biais de scènes
include/pg-scenarioDyn.h
pg-scenarioDyn.cpp
# analyse de scénarios et de ressources multi-scènes
include/pg-scenarioParse.h
pg-scenarioParse.cpp
# Exécution du message OSC
include/pg-script.h
pg-script.cpp
# gestion des capteurs
include/pg-sensor.h
pg-sensor.cpp
# uniformes d'ombrage
include/pg-shader.h
pg-shader.cpp
# Chargement et attribution de textures et de FBO
include/pg-texture.h
pg-texture.cpp
# Réception/émission de messages OSC/UDP
include/pg-udp.h
pg-udp.cpp
# Colle C++
pg-template.hpp
```