

Interpolation sur des points.

Le TP est écrit en C++ multi-plateforme et doit être compilé avec cmake (voir <https://cmake.org/>) et un compilateur C++. Les dépendances sont réduites au minimum, mais pour les expériences il faut un viewer d'images au format BMP (natif sous MS Windows, et utilisez gimp sous Linux).

1. Relire les transparents du cours sur l'interpolation dite de Shepard (« inverse distance », transparents 17-21).
2. Dans ce TP vous interpolerez les données de couleur RGB dans des images. quelques images sont fournies en exemple (cf répertoire « data »), mais vous pouvez en utiliser d'autres, au format bmp 24 bits. Vérifiez que vous pouvez afficher les images à ce format, sinon installez *gimp* ou Paint .NET.
3. Compiler le TP en mode release, voir fichier « compile.txt ».
4. Lancer le programme en ligne de commande, sans aucun paramètre. Parcourez le fichier main.cpp pour comprendre la structure du programme.
5. Parcourez les fichiers point.h et shepard.h (une classe C++ « Point » est fournie avec des fonctions utiles à l'interpolation).
6. Relancer le programme avec la commande « shepard 0 10 ..\..\data\rainbow.bmp out.bmp ». Le second paramètre « power » est la puissance utilisée pour la distance. Relancer avec différents paramètres entre 1 et 100, qu'en déduisez-vous par rapport à la formule des coordonnées barycentriques ?
7. Ajouter dans le fichier « shepard.h » le code pour générer N points aléatoires dans l'image (cf « TODO: add random points »). Recompiler et relancer avec différents points et paramètres de puissance.
8. Ajouter dans le fichier « shepard.h » le code pour générer N points sur une grille régulière dans l'image. Recompilez et relancez avec différents nombres de points et de paramètres de puissance. Quelle est la complexité de l'algorithme en fonction du nombre de pixels de l'image et du nombre de points ?

9. Ajouter le code pour afficher en noir les pixels dans l'image qui correspondent aux points de données.
10. Proposer une méthode de calcul d'erreur moyenne d'interpolation de l'image en entrée : en niveau de gris. Ajouter le code dans le fichier « shepard.h » pour visualiser cette fonction d'erreur dans une image en niveaux de gris.
11. Supposons qu'on se donne un paramètre de tolérance d'erreur maximale : concevoir une méthode itérative (qui ajoute un point après l'autre, ou un paquet de points).
12. Avec un paramètre de tolérance d'erreur maximale : concevoir une méthode hiérarchique capable de construire avec le plus petit nombre de points d'interpolation possible, une interpolation avec une erreur inférieure à cette tolérance.
13. Que proposez-vous comme approximation numérique pour rendre la méthode de calcul de l'interpolation plus locale ?
14. Concevoir une méthode à base d'interpolation dans les triangles d'une triangulation d'une grille régulière.