Diallo Alpha Oumar Binta
(21007631)
Groupe 3.2 IHM

Opérateur getTexel d’accès à un élément de texture

Cette image est de mauvaise qualité car après avoir convertit les coordonnées 2D en une coordonnée 1D, aucun filtrage n'a été effectué, donc il y a perte d'information. On observe un phénomène de pixellisation.
Un exemple serait de prendre une image contenant du texte et de la réduire à une taille inférieure, sans filtrage le texte sera flou sur l'image de sortie et donc illisible.

Opérateur d’interpolation linéaire

Cette opération améliore la qualité de l'image car les valeurs de couleur sont calculées à partir de plusieurs texels pour donner le nouveau pixel. La valeur du pixel final est établie à partir des quatre pixels voisins. L'avantage de cette méthode est sa rapidité mais elle altère un peu l'image car de nouveaux pixels sont créés.

Opérateur d’intégration pour le filtrage de texture

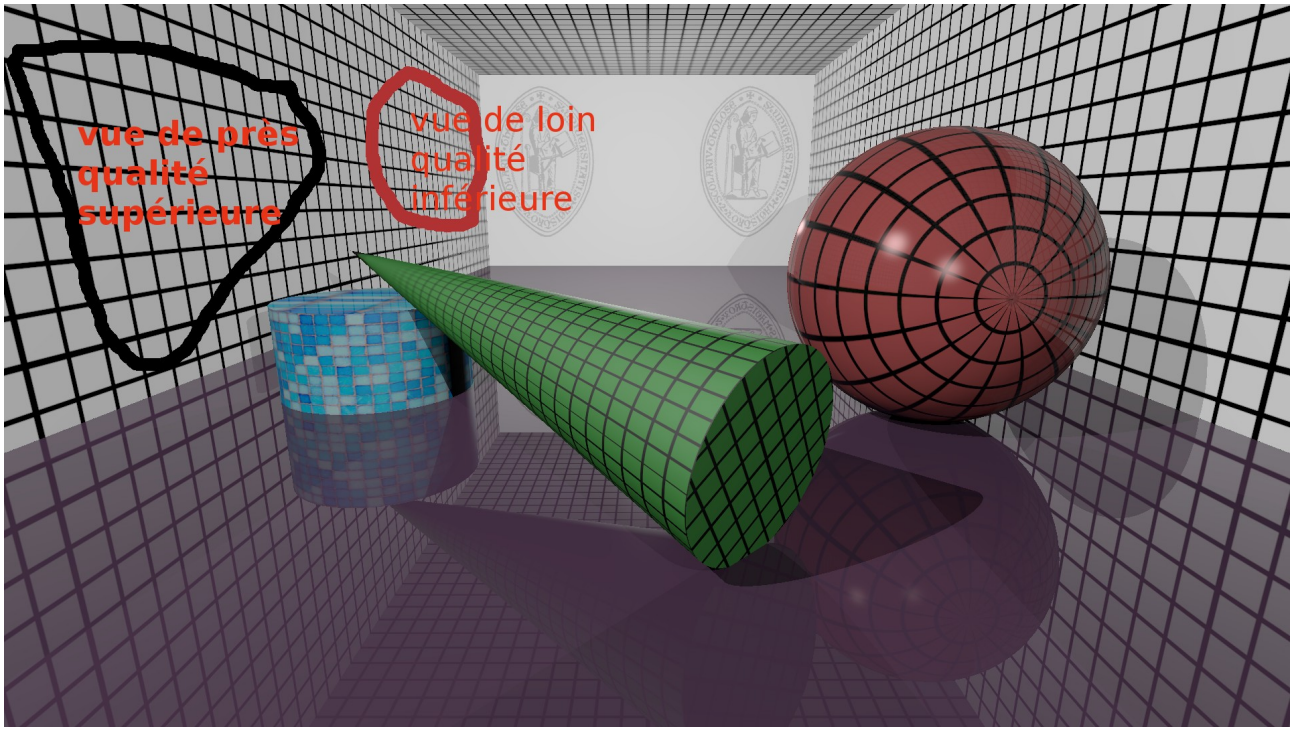
Le filtrage par intégration améliore l'image à un niveau plus élevé que l'interpolation linéaire mais est beaucoup plus coûteuse en temps car la valeur du texel prend la sommation des texels sur le domaine deltas×deltat. Cette opération augmente considérablement le temps de calcul en fonction de la taille de l'image et de la texture appliquée.

Opérateur de pré-filtrage de texture

Le temps de calcul est redevenu acceptable car la résolution des images baisse au fur et à mesure du traitement, elle est divisé par deux par rapport à la précédente. L'image est nettement plus amélioré car chaque pixel d’une image de niveau i correspond à l’intégration de la valeurs de 4 pixels de l’image de rang (i − 1). Les images de rang i sont de meilleures qualités que celle de rang (i-1), donc dans notre scène les objets mis en avant sont de meilleurs qualités que celle du fond (technique de mipmapping).

Analyse et estimation

Le mipmapping est une technique très utilisé dans le domaine du jeu vidéo et de la simulation en temps réel. Elle permet d'éviter la pixellisation lorsqu'une texture est vue de loin. Ici nous nous intéressons aux domaines des jeux vidéos. Le mipmapping permet dans un jeu d'afficher des textures de taille différente selon la distance à laquelle on la regarde. De près on utilise une grande texture détaillée et de loin une petite texture avec très peu de détails.
Supposons un personnage se déplaçant dans un couloir carrelé, à courte distance, on va utiliser par exemple une texture de 256 x 256 pixels pour une apparence réaliste des carreaux et à grande distance on va utiliser une texture de 32 x 32 et sur les distances intermédiaires on va utiliser la 64 x 64 puis la 128 x 128 .



Le mipmapping améliore donc la qualité de la représentation graphique en juxtaposant les versions de la texture diminuées de moitié au fur et à mesure que la profondeur de champ augmente. Dans un jeu vidéo beaucoup de ressources sont utilisés (images, vidéos, sons), tout cela demande beaucoup d'utilisation de la mémoire et un temps de calcul rapide (temps d 'affichage des textures pour la fluidité du jeu). Le mipmapping permet de limiter l'usage de la mémoire de texture de la carte vidéo car lorsqu'elle doit afficher les textures, elle doit les charger dans sa mémoire physique. Le moteur du jeu décide quelles surfaces sont visibles puis envoie à la carte vidéo les textures à mémoriser. Ceci est très important car on peut décider de n'afficher que le côté visible de la surface afin d'économiser de la mémoire; donc cette technique est aussi limité car la qualité du rendu dépend de la mémoire disponible et de la puissance de calcul dont on dispose, les textures les plus

proches sont de meilleurs qualités par rapport à celle plus éloignés (voir figure ci dessus). L’intérêt du mipmapping est très limité si les textures ne sont pas stockées dans la carte vidéo.

Bibiliographie

https://developer.valvesoftware.com/wiki/MIP_Mapping
http://www.adobe.com/devnet/flashplayer/articles/mipmapping.html