Les ombres

DOMON Antoine et MORIN Matthieu

Introduction

Depuis de nombreuses années, diverses techniques permettant le calcul d'ombres voient le jour. Certaines sont plus simples, d'autres plus complexes, certaines sont statiques, d'autres dynamiques, mais toutes ne survivent pas à travers les évolutions des algorithmes. Dans ce document, nous expliquerons les techniques les plus répandues pour le calcul d'ombre dans les jeux, en commençant par les plus simples, et en terminant par les plus complexes. Pour ces techniques, nous expliquerons la théorie, l'utilité ainsi que les points positifs et négatifs.

Les blobs

Les blobs shadow peuvent être considérées comme l'ancêtre des ombres dans le monde des jeux vidéo. Elles permettent de simuler approximativement le comportement d'une ombre sans s'attarder à effectuer des calculs de projection au cours de l'exécution. La technique est simple, il s'agit de dessiner la forme projetée de l'ombre voulue sur une texture, et de l'appliquer sur la surface du terrain sur lequel l'objet auquel on veut l'appliquer se trouve. Il faut donc plaquer la texture aux bonnes coordonnées sur le terrain en fonction de la position de l'objet dans l'espace. Il convient ensuite au programmeur de décider à quel endroit il veut positionner la texture d'ombre en fonction de l'objet. En général, on l'applique de manière à qu'elle se situe sous l'objet en question.

C'est une technique statique car la forme de l'ombre restera invariante dès lors qu'on appliquera des modifications à la géométrie de l'objet, étant donné qu'on l'a pré-dessinée. Cela en fait la méthode la plus simple à implémenter et la plus rapide parmi ses concurrentes malgré les contraintes qu'elle impose. On peut toutefois l'utiliser de manière appropriée dans certaines conditions de luminosité. Par exemple, si on imagine une scène en extérieur avec un soleil de midi,

l'ombre résultante de l'objet va se projeter sous forme de disque aux pied de l'objet, et aura une forme qui aura peu tendance à varier suivant les transformations appliquées à l'objet modèle. Remarquons aussi que le self-shadowing ne sera pas pris en compte avec cette technique, dès lors qu'on dessine seulement l'ombre projetée sur le plan, et qu'on n'applique aucune modification de rendu à l'objet.

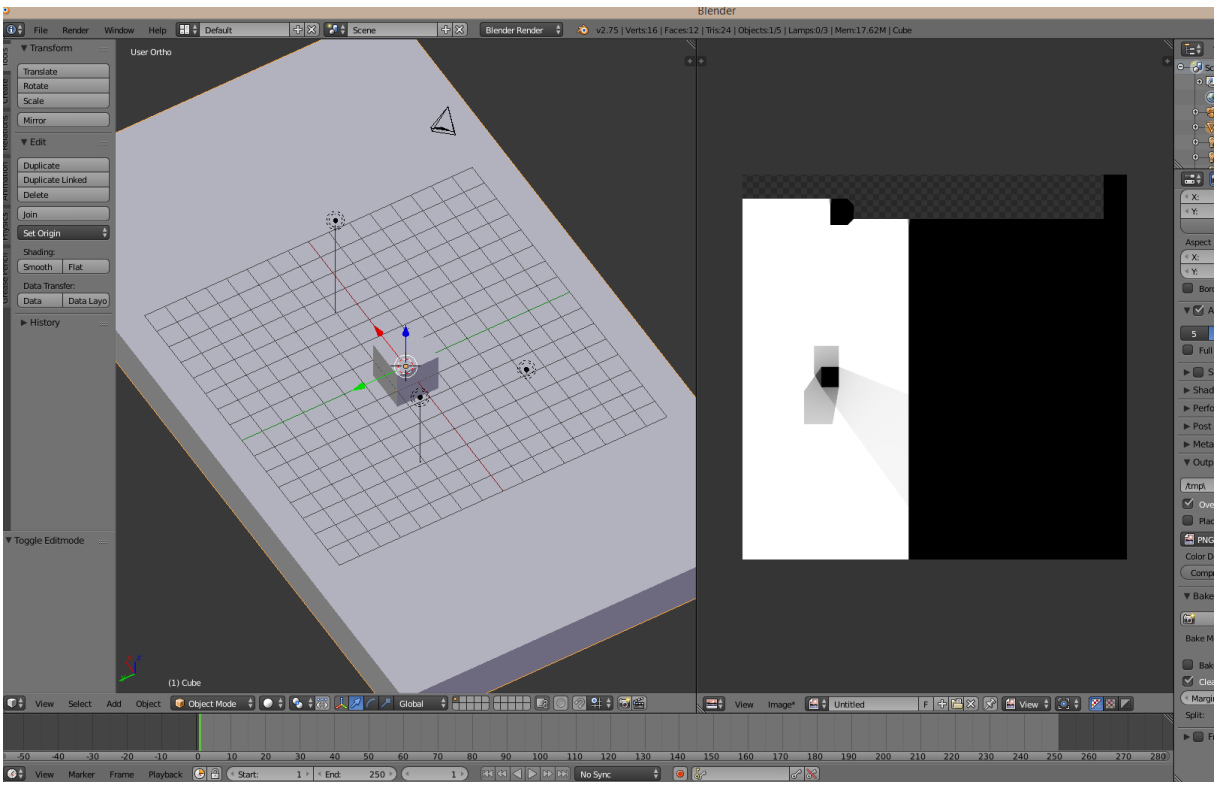
Les matrix shadows

Les matrix shadows constituent une technique simple, rapide et réaliste. Le principe est de calculer pour chacun des points du maillage de l'objet la projection sur le plan sur lequel l'objet se trouve. L'ensemble de ces points projetés constituera notre ombre projetée par l'objet, et on pourra alors lui appliquer les effets de rendu souhaités pour simuler le phénomène d'ombrage.

Cette technique est très rapide étant donné que la projection d'un point sur un plan se fait très rapidement en temps constant (une simple formule mathématique). Elle nous retourne un résultat dynamique étant donné que cette fois-ci, si nous déformons le maillage de notre objet, ou que nous modifions la position ou l'orientation des lumières de notre scène, la projection de l'ombre va se calculer d'une manière différente et nous retournera un résultat cohérent au vu des modifications sur notre scène. Les limites vont alors se poser quand notre terrain aura une forme plus complexe que le plan. Projeter un point sur une forme qui n'est pas définie par un plan est problématique et remet en cause le principe de cette technique pour des scènes compliquées. Aussi, la rapidité de cette méthode sera fixée par la complexité du maillage de l'objet, projeter l'ensemble des points d'une primitive géométrique sera extrêmement rapide, alors que projeter les points d'un maillage complexe de plusieurs milliers de sommets à chaque frame va être un facteur à risque pour la fluidité de l'application. Aussi, comme son homologue plus haut, les matrix shadows ne permettent pas le self-shadowing, ce qui peut lui fait perdre quelque peu en réalisme.

Les light maps / projection de texture

On donne le nom de light map à l'image représentant la carte des luminosités en chaque pixel de l'objet. Plus l'intensité va se rapprocher d'une couleur blanche et alors plus il sera lumineux. Les pixels d'intensité proche du noir correspondront alors à des pixels ombrés sur l'objet. Etant donné un maillage 3D créé dans un éditeur (comme Maya ou Blender), on peut alors générer la light map appropriée à cette objet en fonction de la position et de l'orientation des sources de lumière dans la scène. Voici un exemple d'un cube sur un plan dans l'éditeur Blender, avec à droite le résultat de la génération de la light map.



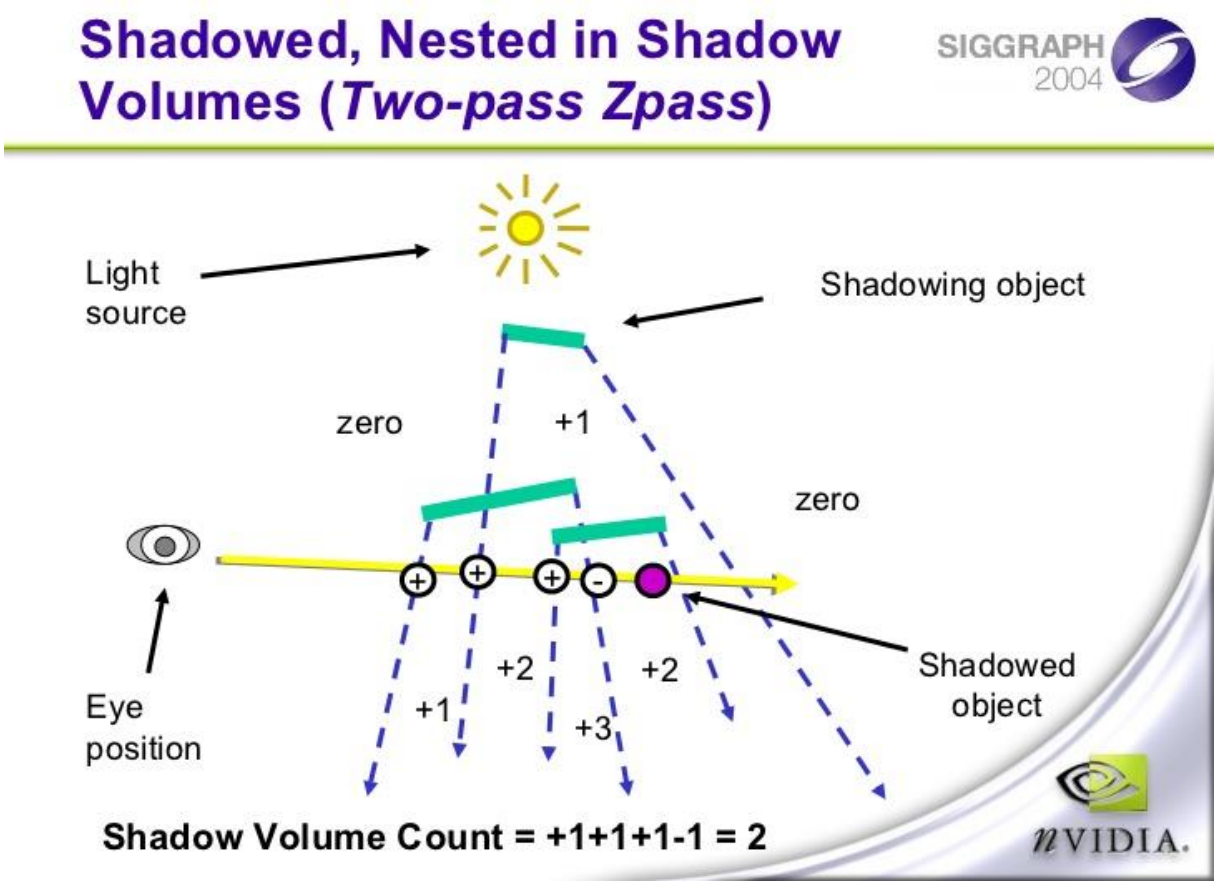
L'image étant pré calculée, il n'y aura pas de temps de calcul supplémentaire au cours de l'exécution, et c'est en cela que réside l'intérêt majeur de son utilisation, d'autant plus que le réalisme peut être poussé si la light map a été bien générée ou bien travaillée par l'artiste. De plus, il est simple de mettre en œuvre le principe, il s'agira simplement de passer l'image à notre Pixel Shader pour déterminer le rendu de chacun des pixels de notre objet. Cependant, la technique montre ses limites lorsque nous disposons d'une scène dynamique. Dès lors qu'on déplacera un objet sur la scène, l'ombre portée dessinée à partir de la light map va rester inchangée et cela va enlever toute cohérence à l'application. On peut dire que c'est une technique qui reste l'une des meilleures et des plus utilisées pour des scènes statiques ou des scènes n'impliquant pas de s'actualiser en temps réel comme celles qu'on peut trouver dans les films d'animation.

Les shadow volumes

L'algorithme de shadow volume utilise le stencil buffer afin de projeter la géométrie sur la scène, en prenant en compte un vecteur de lumière. Cet algorithme fonctionne en « screenspace », c'est-à-dire dans « l'espace écran ». En effet, les ombres ne sont calculées que sur l'écran, et pas sur toute la scène 3D. Deux techniques principales existent. Elles sont appelées Z-Pass et Z-Fail. Le stencil buffer est très pratique pour certains effets graphiques, comme les ombres, le dessin de silhouette, ou même le dessin de contour d'objet.

Le principe du Z-Pass, proposé par Heidmann est plutôt simple à expliquer. On considère que la caméra est dans la lumière. À partir de là, on fait des « lancers de rayon ». Dès qu'on « rentre dans » une extrusion de géométrie, on ajoute 1. De cette manière, on sait que si on est en positif, on est dans l'ombre. De même, dès qu'on « sort » d'une extrusion de géométrie, on soustrait 1. De cette manière, dès lors qu'on est à 0, on sait qu'on est dans la lumière. On comprend donc pourquoi cette technique est faite dans l'espace de rendu, en « screenspace ».
Malheureusement, cet algorithme présente un bug majeur. En effet, on commence par considérer

qu'on est dans la lumière. Or, ça n'est pas forcément vrai. Dans le cas où on est dans l'ombre, le calcul est inversé et est donc faux. En effet, ce qui, pour l'algorithme, devrait être dans l'ombre, est dans la lumière, et inversement.



L'algorithme du Z-Fail, aussi appelé Robust Stencil Shadow Volume, proposé par Carmack, propose un algorithme corrigeant ce bug majeur. Il nous propose d'inverser le calcul, et de projeter les rayons de l'infini, vers la caméra. De cette manière, il n'y a pas de problème d'inversion de l'ombre. Malheureusement, cette correction n'est pas sans conséquences sur la rapidité d'exécution.

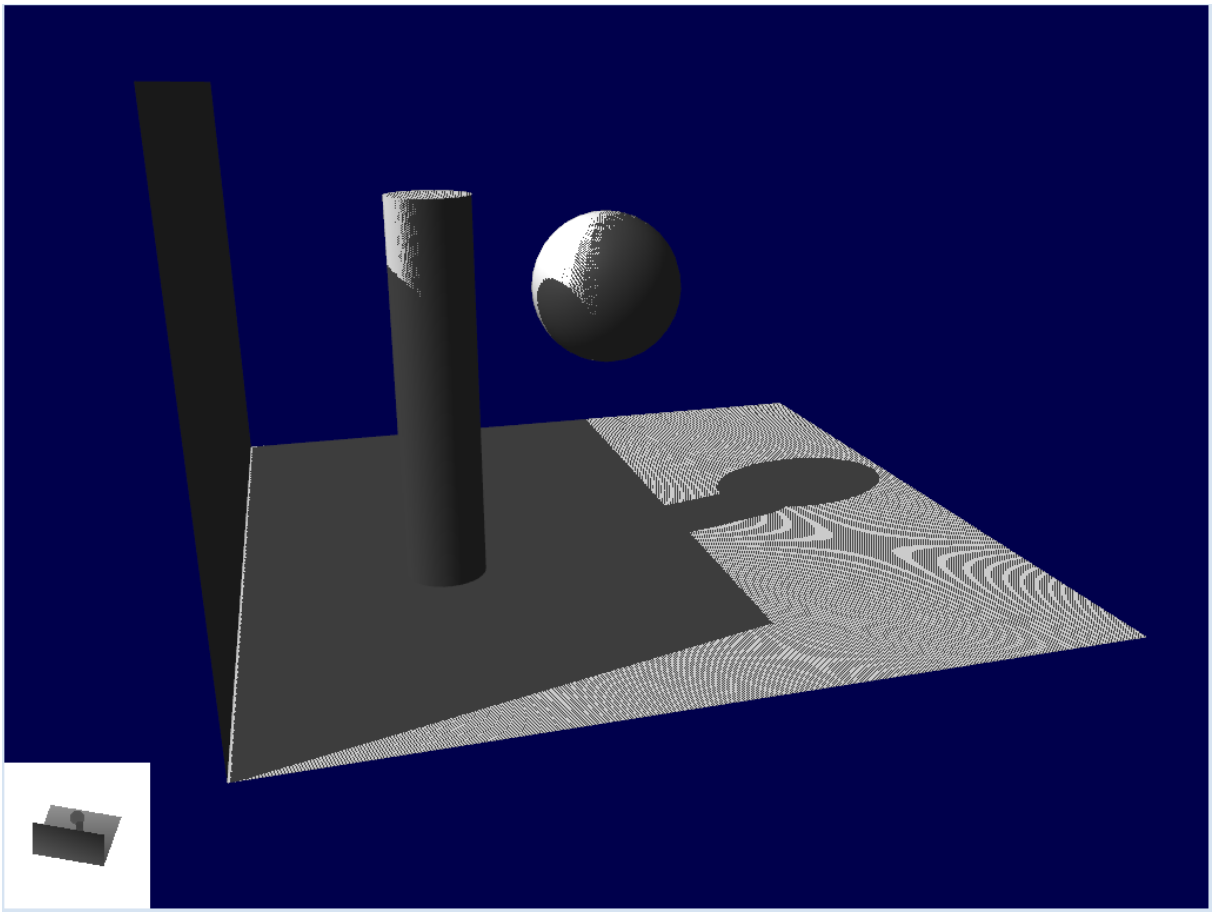
Très peu de jeux utilisent cet algorithme. En effet, en plus d'être assez lourd à exécuter, les améliorations sont peu nombreuses et très complexes. Afficher des ombres floues s'avère être bien plus complexe que pour les shadow maps.

Les shadow maps

Le principe de base des shadow maps consiste à dire que l'ombre est l'image vue par la lumière, projetée sur la scène. Pour cela, on commence par placer la caméra en fonction de la lumière (position et direction), puis on rend une « carte de profondeur ». À partir de cette carte, et en la projetant sur la scène, on peut savoir si un objet est « plus loin » que ce que la « lumière voit ». Si c'est le cas, on sait que notre objet est dans l'ombre. Dans le cas contraire, il est dans la lumière.

De nombreux artéfacts sont observables, sur la technique de base. En effet une carte de profondeur manque de précision. De ce fait, les ombres sont souvent pixellisées, et parfois on peut même observer des artéfacts comme le « shadow acne ». Ce dernier artéfact est principalement causé par le manque de précision de la carte de profondeur. Cependant, il n'est pas le seul ; le projective aliasing est également dû à un problème de manque d'informations. L'algorithme ne peut

pas déterminer si la face est dans l'ombre ou pas car la carte de profondeur est trop imprécise. On a donc le même artéfact que pour le shadow acne. L'autre bug principal, le perspective aliasing, est, lui, dû à une surutilisation d'une donnée dans la carte de profondeur. La carte étant projetée, il arrive qu'un pixel sur la carte de profondeur soit projeté sur une zone trop importante. On peut donc observer une pixellisation sur les bords de l'ombre.



Les années passant, beaucoup d'algorithmes proposent d'améliorer le shadow mapping. Certains proposent de l'adapter aux points de lumière, le shadow mapping étant, à la base, plutôt dédié au calcul de lumière directionnelle. Dans cette catégorie, le plus utilisé est probablement l'algorithme DPSM, pour Dual Paraboloid Shadow Mapping. Cet algorithme permet de dessiner l'ensemble de la scène dans deux textures, dans des coordonnées « paraboloïdes ». Un des artéfacts observés avec cette technique est situé entre les deux textures. En effet, on peut observer une ligne lumineuse qui sépare les deux shadow maps.
D'autres techniques, plus nombreuses, tentent de limiter les artéfacts de pixellisation. Parmi ceux-là, on compte les ESM (exponential shadow map), les PSM (Perspective shadow map), etc…
D'autres tentent d'ajouter au réalisme, en adoucissant les bords, comme le PCF (Percentage Closer Filtering), qui rend l'ombre plus floue, au fur et à mesure qu'on s'éloigne de l'occluder. Une autre, appelée « Soft Shadow Mapping », est plus rapide à exécuter, et rend floue l'ensemble des ombres.
Toutes ces techniques fonctionnent très bien pour des ombres appliquées pour des objets. Cependant, lorsqu'on souhaite calculer de l'ombre pour une scène, tout se complique. C'est là qu'intervient la CSM (Cascaded Shadow Map). Cette méthode propose de découper une scène en zones, plus ou moins proches de la caméra. Pour chacune de ces zones, on calcule une shadow map plus ou moins précise, et, ainsi, on aura des ombres précises près de la caméra, et des ombres moins précises et plus pixellisées loin de cette dernière.