Simulation de flou en synthèse d'images stéréoscopiques pour la réalité virtuelle

Benoît Piranda, François de Sorbier de Pougnadoresse
Institut Gaspard Monge / Université de Marne La Vallée
piranda@univ-mlv.fr / fdesorbi@univ-mlv.fr

Résumé : *Cet article présente une méthode permettant d'ajouter du flou en temps réel aux images de synthèse afin d'assister la perception stéréoscopique en réalité virtuelle. A chaque instant la connaissance combinée du scénario et de la position de l'utilisateur face à l'écran permet de sélectionner automatiquement la zone d'intérêt de la scène. Les éléments nets associés à cette zone attirent l'attention de l'utilisateur tandis que les parties floues évitent une fatigue oculaire excessive.*
Nous présentons tout d'abord différentes méthodes permettant de simuler le flou produit par la présence d'une lentille mince dans les images de synthèse. Puis nous détaillons notre travail sur l'adaptation de ces méthodes au contexte des images stéréoscopiques, illustré par des résultats obtenus sur le système de réalité virtuelle développé dans notre équipe. Enfin, nous évoquons plusieurs solutions permettant de gérer l'interdépendance entre scénario et interactivité dans les animations.

Mots-clés : synthèse d'image temps réel, simulation du flou, réalité virtuelle, scénario, interactivité, stéréoscopie.

1. Introduction

La synthèse d'images stéréoscopiques consiste à produire simultanément deux images de synthèse de la même scène avec des points de vue décalés de façon à simuler l'espace interoculaire de l'utilisateur. Ces images sont ensuite exploitées par un périphérique permettant de transmettre la bonne image à chacun des yeux de l'utilisateur, son cerveau les associant enfin pour interpréter une scène tridimensionnelle.
Les images synthétisées en temps réel sont calculées le plus fréquemment à l'aide d'algorithme de type Z-Buffer travaillant au niveau du pixel pour produire des images parfaitement nettes sur toute leur surface. Cette qualité peut être vue comme un avantage en contexte mono-oculaire mais provoque une fatigue de l'utilisateur, voire une impossibilité de son cerveau de déduire une scène 3D lorsque deux images de stéréoscopie sont localement très différentes.
Le fonctionnement de l'œil humain peut être apparenté à celui d'une lentille mince adaptative capable de rendre nette la zone de la scène sur laquelle l'utilisateur se concentre et de rendre flou le reste de la scène.
Le but de ce travail est de simuler en temps réel la perception des images par l'œil en ajoutant du flou de façon à maintenir l'intérêt de l'utilisateur sur la zone importante de la scène, et réduire la fatigue provoquée par la vision binoculaire d'images de synthèse.
Dans la plupart des scènes de synthèse d'images, il est possible de connaître les éléments de la scène qui doivent intéresser l'utilisateur à un instant donné. Cette information peut être transmise par la position de l'utilisateur face à la scène, ou imposée par un scénario d'utilisation réglant les événements intervenant dans la scène, ou encore combiner scénario et informations sur l'utilisateur.
Dans cet article, nous présentons tout d'abord les règles géométriques régissant la stéréoscopie et la production de flou, ainsi que les méthodes existantes permettant de simuler du flou en images de synthèse. Puis nous détaillons (chapitre 3) notre nouvelle méthode permettant d'adapter l'un de ces algorithmes au contexte de la production des images stéréoscopiques appliquées à la réalité virtuelle. Enfin, le chapitre 4 propose à travers plusieurs illustrations, les règles et outils que nous avons implémenté pour déterminer les éléments importants présents dans les scènes virtuelles.

2. Stéréoscopie et flou

2.1. Principe de calcul des images stéréoscopiques

Une des caractéristiques fondamentales des systèmes de réalité virtuelle [FM01, BC93] est l'immersion de l'utilisateur. Pour cela tous les sens de l'être humain doivent être stimulés au moyen d'outils adaptés. L'utilisation de la projection d'images sur un écran de grande dimension permet de couvrir pleinement le champ de vision de l'observateur mais il subsiste un manque de profondeur qui ne rend pas l'effet d'immersion. Les images stéréoscopiques [Ok76] associent à chaque œil une image d'une même scène mais avec un point de vue différent. A l'aide d'une paire de lunettes filtrantes (filtres polarisants, filtres de couleurs) l'utilisateur fusionne ces deux images afin de créer l'effet de relief. On peut remarquer sur la figure 1 que lorsqu'un élément

de la scène est éloigné devant ou derrière l'écran alors il apparaît à des positions très différentes sur les deux images stéréoscopiques. Sous ces conditions, la reconstruction d'une image en trois dimensions devient fatigante voire impossible.

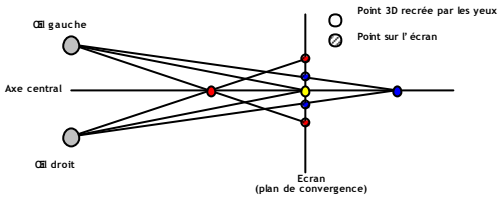

Figure 1: Création d'images stéréoscopiques

2.2.La production du flou

L'ajout de zones floues dans les images de synthèse en réalité virtuelle apporte le double avantage de guider le regard de l'utilisateur vers les zones importantes de la scène (cf. chapitre 4) et de réduire les différences apparentes entre les deux images stéréoscopiques sources de fatigue oculaire.

2.2.1Notion de cercle de confusion

Nous pouvons comparer l'œil humain à une lentille mince capable de faire varier la distance focale *F* afin de se concentrer sur un point précis (phénomène d'accommodation). Les éléments en avant ou en arrière de ce point vont être en dehors du plan focal et créer un cercle de confusion (ou de diffusion) [Fe04, ML00] qui est à l'origine du flou.
La figure 2 présente le schéma de simulation de la vision humaine utilisé par le système d'affichage. La rétine correspond à l'écran de projection, support qui reçoit l'image finale, le cristallin est simulé par une lentille mince placée entre la scène virtuelle et l'écran. Cette lentille mince a pour principe de réfracter les rayons provenant d'un point S de la scène en un point image S'.

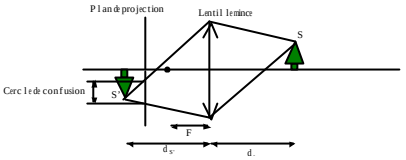

Figure 2 : Le cercle de confusion

La distance focale F correspond à la distance entre le point de convergence des rayons lumineux et la lentille. Nous pouvons caractériser F en fonction de la distance d, du point source *S* et la distance d, de son point image S' par la relation suivante :

$$\frac{1}{F}=\frac{1}{d_{s'}}+\frac{1}{d_s} \qquad (1)$$

On déduit de cette relation, l'expression suivante de la taille C du cercle de confusion, où E représente la taille de la lentille et d, la position du plan de projection par rapport à la lentille :

$$C=\frac{\left|d_{s'}-d_p\right|E}{d_{s'}} \qquad (2)$$

Les équations (1) et (2) montrent comment le choix de la distance focale F va influencer la capacité de réfraction de la lentille et donc la taille du cercle de confusion.

2.2.2 Méthodes de synthèse de flou

Nous classons les méthodes de production d'images de synthèse tenant compte de la profondeur de champ en deux grandes catégories : celles qui réalisent un pré-traitement à la synthèse menant à un suréchantillonnage de l'image, et les méthodes de post-traitement qui dégradent localement une image de résolution classique pour y ajouter du flou.

La première méthode est celle qui respecte le mieux le modèle physique en simulant la propagation de la lumière à travers une ou plusieurs lentilles minces. Elle repose sur l'utilisation d'algorithmes classiques tels que le lancer de rayons distribué [CP84] ou le lancer de cônes permettant un traitement fin du processus de détermination de la couleur d'un pixel. L'algorithme de lancer de rayons distribué permet de suivre la progression des photons dans une scène et de déduire la luminosité d'une zone en fonction de la quantité d'énergie qui l'atteint. En présence d'une lentille mince [MP03], la focalisation est calculée lorsqu'un photon la traverse (cf. figure 3). Le lancer de cône [Am84] est une méthode qui calcule l'aire des objets intersectés par l'ensemble des rayons passant par un pixel, l'énergie reçue par le pixel en provenance de l'objet est déduit proportionnellement à cette surface. Lorsqu'un cône atteint une lentille mince, un nouveau cône de transmission est émis depuis ce point en tenant compte du phénomène de réfraction au niveau de la lentille.

Une autre technique [WN99] plus adaptée à un contexte de synthèse d'images en temps réel consiste à générer plusieurs images pour plusieurs points de vues différents, calculés en tenant compte de l'effet de la lentille. La couleur d'un pixel est obtenue en mélangeant les couleurs calculés pour chaque point de vue. Ce mélange correspond à une moyenne arithmétique implémentée à l'aide du tampon d'accumulation [HA90]. Dans la pratique, la simulation de lentilles minces implique des déplacements très faibles de la caméra, qui induit une variation sur l'image souvent inférieure au pixel. La qualité du flou dépend du nombre de sources énergétiques traitées et donc du nombre d'images accumulés (cf. figure 4). Quoique beaucoup plus rapide que la méthode précédente, cette solution encore trop gourmande en ressources ne peut pas être utilisée en réalité virtuelle (22 images par seconde pour une image d'une résolution de 2048x768 sur un PC BiXeon 2.8Ghz sur une carte vidéo ATI Radeon 9800 PRO).

La seconde catégorie de méthodes repose sur des techniques de post traitement, visant à ajouter du flou à partir d'une image de synthèse nette. Par exemple, la méthode dite de convolution déduit une image floue à partir de deux informations : l'image nette complétée d'une carte de profondeur et d'une matrice de convolution. Pour chaque pixel nous calculons la taille du cercle de confusion à l'aide de la profondeur qui lui est associée [PC82] en utilisant la méthode du paragraphe 2.2.1. La figure 5 présente un résultat obtenu par cette méthode. La qualité est acceptable pour une utilisation en réalité virtuelle mais les temps de calcul sont encore trop important. Par exemple, il faut compter une seconde pour une image de résolution 512x512 sur notre PC de test.

Figure 3: Simulation de lentille mince par lancer de rayon distribué [Tr01]

Figure 4: Exemple de résultat par décalage de la caméra

Figure 5: Résultat obtenu avec l'algorithme de convolution, détails de deux zones produites par des matrices de taille différentes.

2.2.3 Synthèse du flou en temps réel

Pour obtenir du flou en temps réel, une autre technique principalement utilisée dans les jeux vidéo [Ev01, CC01], repose sur l'utilisation de l'image synthétisée représentée à différents niveaux de résolutions. On considère la remarque suivante : une image dont on réduit la résolution apparaît plus floue car elle remplace un ensemble de pixels voisins par un pixel unique de couleur moyenne (cf. figure 6). L'algorithme consiste à calculer tout d'abord toutes les images pour un ensemble de sous-résolutions puis au niveau de chaque pixel, la taille du cercle de confusion déduite de la carte de profondeur comme dans le paragraphe 2.2.1 permet de savoir que l'image doit être sélectionnée.

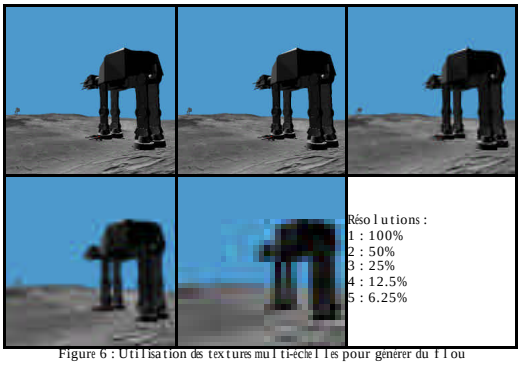

Figure 6 : Utilisation des textures multi-échelles pour générer du flou

Cette méthode a pour avantage d'exploiter deux fonctions temps réel des cartes graphiques actuelles: le calcul des textures multi-échelles [Wil83] (mipmapping en anglais) et la production de la carte de profondeur par l'algorithme du Zbuffer [Ca74].
Les différentes images calculées sont ensuite placées orthogonalement face au point de vue, de façon à recouvrir complètement la zone d'affichage. Plus les images sont éloignées de l'image originale (placée au premier plan), plus elles admettent une résolution basse.

3. Prise en compte du flou dans les images stéréoscopiques

3.1 Notre modèle

Comme nous l'avons présenté au paragraphe 2.1, le système de projection des images stéréoscopique impose de placer le plan de convergence au niveau de l'écran. Dans la plupart des situations, le plan de netteté qui correspond à la zone de la scène sur laquelle se concentre l'utilisateur ne correspond pas au plan de convergence. Or, dans les images stéréoscopiques, la fatigue oculaire est principalement provoquée par l'écart trop important qu'il peut exister entre deux images. Plus une scène visualisée en trois dimensions donne l'impression d'émerger de l'écran ou de s'en éloigner, plus l'utilisateur aura de difficultés à reconstruire cette scène à partir des images stéréoscopiques.
Dans les images stéréoscopiques le plan de convergence est à priori différent du plan de netteté ce qui a pour effet d'imposer des adaptations majeures de la méthode présentée dans le paragraphe 2.2.3. Notre méthode consiste à encadrer le plan de netteté par un ensemble de plans contenant les images de la scène plus ou moins flous en fonction de la distance au plan de netteté comme le montre la figure 7. Les zones susceptibles de provoquer une fatigue oculaire, due aux forts décalages qui peut exister entre deux images stéréoscopiques, sont rendues flous et par conséquent n'attirent plus le regard de l'utilisateur.

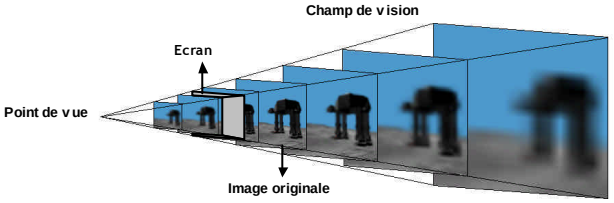

Figure 7 : Exemple de positionnement des plans dans le champ de vision

L'organisation des images de basse résolution autour du plan de netteté est un paramètre important de qualité des images produites. Un placement linéaire peut rendre perceptible le passage d'une image à l'image suivante. Nous avons donc organisé les images suivant une fonction de répartition exponentielle autour de la position du plan de netteté qui réduit les écarts entre les images proches du plan de netteté et augmente les intervalles des images éloignés.

La fonction est de la forme $f(i) = e^{i} * cst * z_0$ où i est le numéro du plan et z_0 la distance entre le plan de netteté et la caméra.

La figure 8 met en évidence la répartition des images de basse résolution autour de la position du plan de netteté (en noir) en les faisant apparaître sous forme de plans à l'aide de niveaux de gris différents. De plus, la maîtrise du nombre d'images affichées à l'écran permet de mieux contrôler la vitesse d'exécution du logiciel

Figure 8: Affichage des plans en niveaux de gris

3.2 Algorithme

L'algorithme de simulation du phénomène de flou se déroule en deux passes :

Première passe

1. Rendre la scène et déterminer la position p du plan de netteté.
2. Sauvegarder le contenu du tampon chromatique dans une texture et créer l'ensemble des mipmaps associées.
3. Vider le tampon chromatique.

Deuxième passe

1. Désactiver l'écriture dans le tampon de profondeur et redéfinir la fonction de comparaison pour ne conserver que les fragments ayant une valeur de profondeur plus élevée que celle contenue dans le tampon de profondeur.

2. Créer $m + n + 1$ plans recouvrant l'écran où m est le nombre de plans situés derrière le plan de netteté et n le nombre de plan devant.
3. Attribuer à chaque plan une mipmap en fonction de la taille du cercle de confusion qui lui est associé.
4. Dessiner les plans du plus éloigné au plus proche de la caméra avec le plan de netteté situé à une distance p de la caméra.

Pour chaque pixel le choix du plan à afficher est déterminé par le contenu du tampon de profondeur. Comme présenté dans la figure 9 : si pour un pixel donné la valeur contenue dans le tampon de profondeur se situe dans le volume V alors le plan P sera affiché.

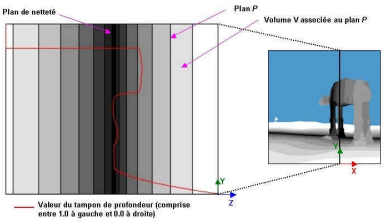

Figure 9 : Lien entre le tampon de profondeur et les plans

3.3 Résultats

La figure 10 présente un résultat obtenu en appliquant notre méthode de génération du flou que nous venons de décrire. Nous avons un affichage de 24 images par seconde en utilisant notre scène test (environ 70.000 facettes) avec une résolution de 1024x768 stéréo sur un BiXeon 2,8Ghz avec une ATI Radeon 9800 PRO.

Figure 10 : Exemple de résultat obtenu pour la simulation de flou

4 Zone d'intérêt dans les scènes de réalité virtuelle

La zone d'intérêt correspond à la position dans l'espace de la scène virtuelle sur laquelle se porte l'intérêt de l'utilisateur. Dans un contexte cinématographique considérant un utilisateur passif devant la scène, cette position peut être réglée par le scénario, il correspond au lieu où se déroule l'action. Dans un contexte de pure interactivité où l'utilisateur est libre de se promener dans un environnement virtuel, la zone d'intérêt peut être placée à une distance fixe face à l'observateur (distance qui peut varier en fonction de la vitesse de déplacement

de l'utilisateur qui a tendance à anticiper les obstacles en portant son regard plus loin lorsqu'il se déplace rapidement).

D'une manière plus générale, les logiciels multimédia tels que les jeux ou les logiciels de simulation qui exploitent ces environnements reposent sur une situation intermédiaire où l'utilisateur admet un comportement relativement libre mais est confronté à des événements qui l'interrompent. Nous proposons un scénario capable de traiter ces différentes phases en jouant sur la position du plan de netteté.

La gestion du plan de netteté par le scénario permet de sélectionner à chaque instant la zone importante de la scène et permet une communication explicite de cette information à l'utilisateur. A chaque instant nous pouvons déterminer dans un script temporel l'objet le plus important de la scène en plaçant le plan de netteté à son niveau.

4.1.Positionnement dirigiste

Lorsque nous voulons donner de l'importance à certains éléments composant une scène, il est possible d'intervenir sur l'algorithme de rendu durant le scénario de déroulement de l'action. A un instant donné, le scénario modifie les caractéristiques géométriques (forme, position, taille, direction, déplacement...) et radiométriques (couleur visible, caméra, lumière...) de ces objets afin d'obtenir le résultat visuel désiré. Ceci est fait à l'aide d'un fichier "script" énumérant toutes les commandes à exécuter à chaque moment clé du scénario.

Figure 11 : exemple de scénario dirigiste

L'exemple illustré figure 11 présente une animation futuriste durant laquelle l'utilisateur suit un engin du regard. Le script correspondant, dont un extrait est présenté ci-dessous, gère les événements qui s'y produisent. Ce scénario déclenche à un instant donné un événement particulier : nous pouvons observer deux projectiles se diriger sur l'engin. La réaction naturelle de l'observateur est de chercher la provenance de ces projectiles. Cette réaction est intégrée au scénario en positionnant le plan de netteté sur l'agresseur durant toute la séquence suivante.

```
temps 0
// définition des paramètres du robot
positionne objet 1 vecteur [30.0,30.0,  -30.0]
affiche objet 1
nettete objet 1
stoppe objet 1
// définition des paramètres du vaisseau
trajectoire objet 0 courbe  0
vitesse objet 0 reel 100.0
bouge objet 0
affiche objet 0
// définition des paramètres du projectile
positionne objet 2 vecteur [0.0,0.0,0.0]
vitesse objet 2 réel 1000.0
stoppe objet 2
enleve objet 2
```

```
temps 20000
// Le vaisseau tire le projectile
// La trajectoire et la position d'origine
// du projectile sont définies dans l'action
demarre action 1 objet 0
affiche objet 2
bouge objet 2
temps 30000
nettete objet 2
temps 50000
nettete objet 0
temps 80000
// Le vaisseau retire le projectile
demarre action  1 objet 0
temps 100000
// Le script est redémarré
redemarre
```

4.2.Positionnement interactif

Une gestion moins dirigiste du plan de netteté est possible en adaptant celle-ci aux conditions de position et d'orientation de l'utilisateur dans la scène virtuelle. Par exemple, l'objet sélectionné peut être celui placé dans la direction d'observation de l'utilisateur. Ou bien, cette sélection peut tenir compte de paramètres morphologiques des objets dans l'image, par exemple l'utilisateur aura tendance à se concentrer sur les objets les plus gros représentés dans l'image.

Le contexte de visites d'un musée virtuel où plusieurs œuvres sont présentées permet de mettre cette catégorie de navigations en évidence. Le script permet ici de régler la position du plan de netteté sur une œuvre sélectionné automatiquement en fonction de la position du visiteur dans la salle (cf. figure 12).

Figure 12 : Exemple de navigation dans le musée virtuel.

5.Conclusion

Nous avons présenté dans cet article, une nouvelle solution permettant à la fois de mettre en évidence des zones considérées comme importantes dans la scène de synthèse d'images temps réel utilisés en réalité virtuelle et de réduire la fatigue oculaire produite par les différences fortes pouvant apparaître entre deux images stéréoscopiques.
La sélection de ces zones d'intérêt auxquelles est associé le plan de netteté peut correspondre à l'objet placé face à l'utilisateur, ou bien dépendre de caractéristiques (morphologiques, chromatiques, d'intérêt...) des objets présents dans la scène, ou encore être imposée par un scénario qui décrit le déroulement des actions dans la scène.
Ce travail n'est qu'une étape dans la gestion combinée d'un scénario et des informations sur l'utilisateur pour l'immersion multi-sensorielle dans les scènes de réalité virtuelle. Une simulation similaire dans le domaine sonore doit permettre d'encore mieux immerger l'utilisateur dans une ambiance 3D complète.

Références

[Am84] J. Amanatides, "Ray-Tracing with Cones", Computer Graphics, 18, 3, (1984), pp. 129-134.

[BC93] Grigore Burdea and Philippe Coiffet. *La Réalité Virtuelle*. Hermes, 1993.

[Ca74] E. E. Catmull, A subdivision algorithm for computer display of curved surfaces. PhD thesis. University of Utah, December 1974.

[CC01] R. Cant, N. Chia and D. Al-Dabass. New anti-aliasing and depth of field techniques for games. The Second International Conference on Intelligent Games and Simulation, London, November 30 : 117-122, 2001.

[CP84] R. L. Cook, T. Porter, and L. Carpenter, Distributed Ray Tracing. *Computer Graphics*, vol. 18, no. 3, pp 137-145, 1984.

[Ev01] A. Evans. Four Tricks for Fast Blurring in Software and Hardware. http://www.gamasutra.com/features/20010209/evans_01.htm : 3, 2001.

[Fe04] R. Fernando. GPU Gems : Programming Techniques, Tips and Tricks for Real-Time Graphics. Addison-Wesley Professional, 2004.

[FM01] Philippe Fuchs, Guillaume Moreau, and Jean-Paul Papin. Le traité de la réalité virtuelle. *Les Presses de l'Ecole des Mines*, 2001. pages 330-333, 350-358.

[HA90] Paul E. Haeberli and Kurt Akeley, The Accumulation Buffer: Hardware Support for High-Quality Rendering. *Computer Graphics*. 24(4), 309– 318, August 1990.

[ML00] J. Mulder and R. van Liere. Fast Perception-Based Depth of Field Rendering. Procedings of the ACM symposium on Virtual reality software and technology, Seoul, Korea : 129-133, 2000.

[MP03] S. Michelin and C. Pichard. La chambre photographique, RCFAO, Volume 18, Edition Lavoisier, Hermè Science : 231-245, 2003.

[Ok76] T. Okoshi. Three-dimensional Imaging techniques. Academic Press. Inc., New York, 1976.

[PC82] M. Potmesil and I. Chakravarty. Synthetic image generation with a lens and aperture Camera Model. ACM Transactions on Graphics (TOG), 1(2) : 85-108 ,1982.

[Tr01] B. Trichet, "Simulation informatique des caractéristiques optiques d'une chambre photographique et de ses composants", mémoire de recherché, Ecole Nationale supérieur Louis Lumière, 2001.

[Wi83] L. Williams, Pyramidal parametrics. Computer Graphics. 17(3), pages 1–11, july 1983.

[WN99] M. Woo, J. Neider and T. Davis. OpenGL 1.2. CampusPress, 1999.