جامعة سيدي محمد بن عبد الله بفاس
UNIVERSITÉ SIDI MOHAMED BEN ABDELLAH DE FES

كلية العلوم ظهر المهراز فاس
FACULTE DES SCIENCES DHAR EL MAHRAZ FES

N° d'ordre : 98/2023

Centre des Etudes : ***Sciences et Techniques et Sciences Médicales***

Formation Doctorale : ***Sciences et Technologies de l'information et de communication***

Discipline : ***Informatique***

Spécialité : ***Informatique***

Laboratoire : ***Laboratoire d'Informatique, Signaux, Automatique et Cognitivisme***

THÈSE DE DOCTORAT

NOUVEAUX ALGORITHMES DE PLACAGE DE DEPLACEMENT TRIDIMENSIONNEL PAR PIXEL AVEC TRAITEMENT DE LA SILHOUETTE

Présentée par :

OUAZZANI CHAHDI Adnane

Soutenue le 16/12/2023 devant le jury :

	Nom et Prénom	Grade	Etablissement
Président	TAIRI Hamid	PES	Faculté des sciences Dhar el Mahraz- Fès
Directeur de thèse	SATORI Khalid	PES	Faculté des sciences Dhar El Mahraz- Fès
Co-Directeur de thèse	HALLI Akram	PH	Faculté des Sciences Juridiques Economiques et Sociales – Meknès
Rapporteurs	OUANAN Mohammed	PES	Faculté des Sciences Université Molay Ismail Mekhnès
	El MOUBTAHIJ Hicham	PH	EST, Université Ibn Zohr Agadir
	El ABDERRAHMANI Abdellatif	PH	Faculté des sciences Dhar El Mahraz- Fès
Membres	MERRAS Mostafa	PH	EST, Université Molay Ismail Mekhnès
	El AKKAD Nabil	PH	Ecole Nationale des Sciences Appliquées de Fès
	YAHYAOUI Ali	PES	Faculté des sciences Dhar El Mahraz- Fès

Année universitaire : 2022-2023

NOUVEAUX ALGORITHMES DE PLACAGE DE DEPLACEMENT TRIDIMENSIONNEL PAR PIXEL AVEC TRAITEMENT DE LA SILHOUETTE

Le placage de déplacement tridimensionnel par pixel est une technique prometteuse en infographie 3D qui vise à améliorer la qualité visuelle des scènes 3D sans modifier leur géométrie de base. Cette technique permet de simuler les microreliefs présents sur les objets 3D en utilisant les techniques de placage de texture et les algorithmes de traçage de rayons. En réduisant le nombre de primitives graphiques du maillage 3D, cette technique optimise le temps d'exécution et l'occupation mémoire tout en préservant le réalisme des scènes et l'interactivité avec l'utilisateur.

Adnane OUAZZANI CHAHDI

16/12/2023

DEDICACE

Je dédie ce travail à :

Ma mère, source puissante de tendresse et d'amour, celle qui m'a appris l'art de la patience, pour son soutien tout le long de mon parcours.

Mon père, qui m'a toujours soutenu avec volonté et amour et qui m'a toujours fait des sacrifices pour m'aider.

Mon épouse Amina ARBAH, depuis que je t'ai connue, tu n'as cessé de me soutenir et de m'épauler. Tu me voulais toujours le meilleur.

Mes chers enfants Aicha et Salmane, vous êtes ma plus grande fierté et ma source d'inspiration infinie.

Mes frères, qui m'ont montré le vrai sens de la fraternité, pour leurs encouragements et appui sans faille.

Ma chère grande famille.

Mes chers amis et enseignants.

Qu'ALLAH leur accorde santé et prospérité.

Adnane

REMERCIEMENT

Nos sincères remerciements vont tout d'abord à notre seigneur le plus miséricordieux pour sa grâce et sa bénédiction.

Je tiens ensuite à exprimer mes gratitudes envers mon directeur de thèse Pr. Khalid SATORI qui m'a offert la possibilité de réaliser cette thèse et a accepté de diriger mes travaux de recherches. Ce travail est une occasion pour moi d'apprécier mes qualités humaines et professionnelles.

Je remercie aussi vivement Monsieur Akram HALLI, mon encadrant, qui m'a enrichi par sa vision critique, ses orientations et ses conseils pertinents. Je voudrai aussi le remercier pour tout le temps et la patience qu'il m'a accordés. Permettez-moi Mr. de vous exprimer mon profond respect.

Mes vifs remerciements vont aussi à tous les membres de jury, Je vous suis reconnaissant pour l'amabilité avec laquelle vous avez accepté de juger mon travail.

Je remercie spécialement mon ami Anouar Ragragui, pour notre étroite collaboration tout au long de nos études doctorales.

Mes remerciements vont également au corps professoral qui a toujours veillé sur le déroulement efficace de mes études.

À mon épouse Amina ARBAH, je suis infiniment reconnaissant pour ton amour inconditionnel, ton soutien indéfectible et ta présence constante dans ma vie, sans lesquels cette thèse n'aurait pas été possible.

Je ne saurais terminer ces remerciements sans exprimer notre gratitude envers tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce travail.

RESUME

Les progrès fulgurants du matériel informatique durant ces dernières années témoignent de la volonté de suivre l'évolution grandissante de l'infographie qui se réponde dans divers domaines d'applications. En effet, les scènes 3D créées par ordinateur représentent depuis longtemps un véritable défi dans le domaine de l'infographie, en particulier lorsqu'il s'agit de générer des rendus en temps réel de scènes complexes comportant des surfaces très riches en détails. Bien que d'importants progrès aient été accomplis dans ce contexte, la demande de réalisme n'a pas cessé d'augmenter. D'où l'intérêt de nouveaux algorithmes permettant d'optimiser le temps d'exécution et l'occupation mémoire, tout en conservant la qualité visuelle globale de la scène.

Pour répondre aux exigences croissantes en matière de réalisme et d'interactivité dans les applications d'infographie 3D, des techniques prometteuses consistent à utiliser le placage de déplacement tridimensionnel par pixel. L'objectif est de réduire le nombre de primitives graphiques constituant le maillage 3D (sommets et polygones) en se basant sur les techniques de placage de texture et les algorithmes de traçage de rayons. Ces techniques reposent sur des approches révolutionnaires pour enrichir l'apparence visuelle des scènes 3D en simulant les microreliefs présents sur une variété d'objets 3D sans pour autant modifier leurs géométries. En adoptant les meilleures techniques du placage de déplacement tridimensionnel par pixel, il est possible d'améliorer davantage les performances du matériel et de répondre efficacement à la demande croissante de réalisme et d'interactivité dans les applications d'infographie 3D.

Dans le cadre de ce travail, nous nous sommes intéressés aux trois meilleures techniques de la littérature, à savoir : le placage de relief, le traçage de cônes et le placage d'extrusion. Bien que ces méthodes offrent des rendus proches du réalisme avec des coûts relativement faibles, la qualité de certaines images dépend étroitement du nombre d'itérations de l'algorithme de traçage de rayons. De plus, certaines images souffrent du manque de la prise en charge de la silhouette sur les contours des objets 3D.

Pour remédier à ces limites, nous avons proposé de nouvelles approches. En ce qui concerne le placage de relief, nous avons utilisé une synchronisation entre le nombre d'itérations, le facteur d'échelle des microreliefs et l'angle de vue. Ceci garantit que les objets 3D conservent leur réalisme quelles que soient les conditions et contraintes inhérentes au processus du rendu 3D. Pour la technique de traçage de cônes, nous avons introduit une troisième version qui agit comme un interpolateur entre la version conservative et la version relaxée. Ce faisant, nous capitalisons sur leurs avantages tout en réduisant leurs inconvénients respectifs.

Enfin, nous avons proposé une nouvelle approche pour la prise en charge de la silhouette, aussi bien pour la technique de traçage de cônes que pour celle du placage d'extrusion. Nous avons ainsi formulé une modélisation quadratique au niveau de chaque sommet du maillage 3D. Cette modélisation est calculée lors d'une phase de prétraitement et est représentée par deux paramètres réels pendant la phase de rendu. Ces deux techniques se trouvent nettement améliorées et augmentent ainsi sensiblement le réalisme des micro détails grâce à la gestion des contours.

Mots-Clés : *rendu temps réel, mesostructures, placage de déplacement par pixel, traçage de cônes, placage de relief, plage d'extrusion, traitement de la silhouette, approximation quadratique.*

Abstract

New Algorithms for Per-pixel Displacement Mapping with Silhouette Processing

The rapid progress of computer hardware in recent years shows a desire to follow the growing evolution of computer graphics that is used in various application areas. However, it has become evident that material performance alone is not enough to satisfy the increasingly demanding requirements in terms of quality and interactivity. Indeed, computer-generated 3D scenes have long been a real challenge in computer graphics, especially when it comes to generating real-time renderings of complex scenes with very detailed surfaces. Despite of the important progress that has been accomplished in this context, the demand for realism has never stop growing, which express the need of new algorithms to optimize execution time and memory occupation, while maintaining the overall visual quality of the scene.

To meet the increasing demands for realism and interactivity in 3D computer graphics applications, promising techniques consist of using per-pixel displacement mapping which aims to reduce the number of graphics primitives of the 3D meshes (vertices and polygons) based on texture mapping techniques and ray-tracing algorithms. These techniques are based on revolutionary approaches to enrich the visual appearance of 3D scenes by simulating the microreliefs present on a variety of 3D objects without modifying their geometries. By adopting the best techniques of per-pixel displacement mapping, it is possible to improve the hardware performance and effectively to deal with the growing demand of realism and interactivity in 3D computer graphics applications.

As part of this work, we were interested in the three best techniques in the literature, namely: relief mapping, cone tracing and extrusion mapping. Although these methods provide renderings close to realism at relatively low costs, the quality of some images is closely tied to the number of iterations of the ray tracing algorithm. Additionally, some images suffer from the lack of silhouette support on the edges of 3D objects.

To address these limitations, we have proposed new approaches. Regarding the relief mapping, we used a synchronization between the number of iterations, the microreliefs scale factor and the viewing angle. This ensures that 3D objects maintain their realism under any conditions and constraints inherent to the 3D rendering process. For the cone tracing technique, we introduced a third version that acts as an interpolator between the conservative and relaxed versions. In doing so, we capitalize on their advantages while reducing their respective disadvantages.

Finally, we proposed a new approach for silhouette support, both for the cone tracing technique and for the extrusion mapping technique. We have thus formulated a quadratic modeling at each vertex of the 3D mesh. This modeling is computed during a preprocessing phase and is represented by two real parameters during the rendering phase. Both of these techniques have been significantly improved, substantially enhancing the realism of micro-details through edge handling.

Keywords: *real-time rendering, mesostructures, per-pixel displacement mapping, cone tracing, relief mapping, extrusion range, silhouette processing, quadratic approximation*.

TABLE DES MATIÈRES

Introduction Générale ——— 17

Présentation du sujet ---- 18
Cadre de la thèse ---- 19
Motivations et But ---- 20
Contributions ---- 21
Environnement logiciel et configuration matérielle ---- 22
Organisation du document ---- 23

1 Fondements de base de l'infographie 3D ——— 25

1.1 Introduction ---- 26
1.2 Historique ---- 27
1.2.1 Les années 1920-1960 ---- 27
1.2.2 Les années 1960-1980 ---- 28
1.2.3 Les années 1980-2000 ---- 30
1.2.4 Les années 2000 à aujourd'hui ---- 32
1.3 Création des scènes 3D ---- 33
1.3.1 Conception est modélisation ---- 34
1.3.2 Le rendu ---- 37
1.4 Pipeline graphique ---- 39
1.4.1 Préparation des données et prétraitement ---- 41
1.4.2 Assembleur des entrées ---- 42
1.4.3 Transformation des sommets ---- 42
1.4.4 Rastérisation ---- 45
1.4.5 Transformation des pixels ---- 46
1.4.6 Mélange de couleurs et affichage ---- 47
1.5 Domaines d'application ---- 47
1.6 Conclusion ---- 51

2 Modèles d'ombrage et placage de textures ——— 52

2.1 Introduction ---- 53
2.2 Structures des objets 3D ---- 53
2.3 L'éclairage dans les scènes 3D ---- 55
2.3.1 Les couleurs ---- 55
2.3.2 Les sources de lumière ---- 56
2.3.3 Les lumières directionnelles ---- 56
2.3.4 L'illumination ---- 58
2.4 L'ombrage ---- 60
2.4.1 Les composants de l'ombrage ---- 60
2.4.2 Modèles d'ombrage ---- 62
2.5 Placage de texture ---- 64
2.5.1 Principe ---- 64
2.5.2 Les cartes d'éclairage ---- 65
2.5.3 Placage de bosselures ---- 65

2.5.4 Placage de normales---67
2.6 Placage de déplacement par sommet --- 69
2.6.1 Déplacement des sommets ---69
2.6.2 Densification du maillage 3D---70
2.7 Placage de déplacement par pixel --- 71
2.7.1 La carte de déplacement---71
2.7.2 Traçage de rayons---72
2.8 Conclusion --- 73

3 Les travaux réalisés en : Techniques de placage de déplacement par pixel---74

3.1 Introduction--- 75
3.2 Recherche du point d'intersection --- 76
3.2.1 Techniques sans encodage de l'espace vide---77
3.2.2 Technique avec encodage de l'espace vide ---82
3.3 Traitement de la silhouette--- 88
3.3.1 Représentation locale ---88
3.3.2 Placage à l'aide de coquille---90
3.3.3 Placage de déplacement dépendant du point de vue ---90
3.4 Auto-ombrage --- 91
3.4.1 Placage des ombres ---92
3.4.2 Placage d'horizon---92
3.4.3 Auto-ombrage par carte de visibilité---93
3.4.4 Fonction d'illumination globale ---93
3.5 Conclusion --- 94

4 Placage dynamique de déplacement par pixel avec traitement de la silhouette---95

4.1 Introduction--- 96
4.2 Placage dynamique de relief --- 97
4.2.1 Placage de relief---97
4.2.2 Traçage dynamique de rayon---98
4.3 Technique de traçage de cônes hybrides ---101
4.3.1 Les techniques conservatives et relaxées --- 101
4.3.2 La technique hybride--- 101
4.4 Technique de traçage de cônes avec pris en charge de la silhouette ---102
4.4.1 L'approximation quadratique --- 103
4.4.2 Traçage de cônes avec correction de la silhouette --- 107
4.5 Résultats et discussion ---114
4.5.1 Synchronisation combinée --- 114
4.5.2 Cônes hybrides --- 114
4.5.3 Traitement de la silhouette --- 115
4.6 Conclusion ---121

5 Placage d'extrusion par pixel avec traitement de la silhouette----123

5.1 Introduction---124
5.2 Placage d'extrusion par pixel ---124

5.2.1 La carte de forme ---- 125
5.2.2 Le traçage de rayons ---- 125
5.2.3 Les types de placage d'extrusion par pixel ---- 126
5.3 Placage d'extrusion par pixel avec correction de la silhouette ---- 128
5.3.1 Rectification du rayon de vue ---- 128
5.3.2 Rectification de la distance ---- 130
5.3.3 Algorithme de placage d'extrusion avec correction de la silhouette ---- 131
5.4 Résultat et discussion ---- 134
5.5 Conclusion ---- 138

Conclusion et Perspectives ---- 139

Bilan ---- 140
Perspectives ---- 142

Publications ---- 144

Annex A : Généralités sur l'infographie ---- 145

Annex B : Organigrammes ---- 152

Annex C : Lexique Français/Anglais ---- 155

Bibliographie ---- 156

LISTE DES FIGURES

Figure 1.1 : (a) Premier ordinateur à affichage graphique Whirlwind I. (b) Les premières images de Ben Laposky. ---- 28

Figure 1.2 : (a) Capture d'écran du jeu *Spacewar*. (b) Le mini-ordinateur DEC PDP-1. (c) Console de jeux *Odyssey*. ---- 29

Figure 1.3 : (a) Système Sketchapad. (b) Premier système CAO de General Motors et IBM. ---- 29

Figure 1.4 : (a) Capture d'écran du premier film d'animation 3D. (b) Premier système de réalité virtuelle. (c) Animation 3D créé par ordinateur [9]. ---- 30

Figure 1.5 : Modélisation 3D de la théière la plus célèbre au mode de l'infographie 3D [10]. ---- 30

Figure 1.6 : (a) Carte d'affichage vidéo MDA. (b) Carte d'affichage vidéo HGC. ---- 31

Figure 1.7 : (a) Carte graphique RIVA 128. (b) Carte graphique GeForce 256. ---- 31

Figure 1.8 : Les premières cartes graphiques programmables. (a) Programmation limitée. (b) Programmation avancée. ---- 32

Figure 1.9 : Logiciels de modélisation et de rendu 3D. Blender (à gauche) et Autodesk Maya (à droite). - 35

Figure 1.10 : Exemples de maillages polygonaux construits en utilisant des sommets, des triangles, des polygones, des courbes, des surfaces, etc. ---- 35

Figure 1.11 : Objets 3D créés à l'aide de différentes techniques. (a) Déplacement des sommets. (b) Extrusion des surfaces. (c) Révolution d'une courbe. ---- 35

Figure 1.12 : Application des textures et des couleurs a une sphère. (a) Placage de texture. (b) Application des matériaux. ---- 35

Figure 1.13 : Mise en scène des différents objets 3D. (a) Les objets sans utilisés directement dans la scène. (b) Les objets ont subis des opérations de prétraitement. ---- 36

Figure 1.14 : Exemples de deux caméras représentants deux types de projections. A gauche, une caméra avec une projection perspective. A droite, une caméra avec une projection orthogonale. ---- 36

Figure 1.15 : Exemples d'animation 3D. (a) Animation de personnages. (b) Simulation des éffets phisyques et des phénomènes naturels. ---- 36

Figure 1.16 : Scènes contenant des objets 3D et des sources de lumière. ---- 37

Figure 1.17 : Rendu basé sur la physique. ---- 37

Figure 1.18 : Rendu d'une scène composée de différents objets 3D. (a) Rendu avec placage de texture seulement. (b) Rendu avec placage de texture et en appliquant les techniques d'illumination. ---- 38

Figure 1.19 : Exemples de rendu de deux scènes 3D. (a) Rendu précalculé. (b) Rendu en temps réel. ---- 39

Figure 1.20 : Les grandes étapes du pipeline graphique. Les étapes programmables sont schématisées en jaune. ---- 40

Figure 1.21 : Différentes primitives graphiques d'OpenGL pour connecter les sommets du maillage 3D. 42

Figure 1.22 : Représentation d'une théière 3D dans différents espaces. (a) Espace local. (b) Espace globale. (c) Espace de vue. ---- 44

Figure 1.23 : Le volume délimité par les plans de la figure est appelé volume de vue. ---- 45

Figure 1.24 : Processus de transformation d'un objets 3D de son espace local à l'espace écran. ---- 45

Figure 1.25 : Jeux et films d'animations populaires crées avec les techniques modernes de l'infographie 3D. ---- 48

Figure 1.26 : Exemples d'architectures et de conception crées par ordinateur. ---- 49

Figure 1.27 : Représentation des organes du corps humain en utilisant l'infographie 2D/3D. ---- 49

Figure 1.28 : Utilisation de l'infographie dans les domaines de l'éducation et la science. ---- 49

Figure 1.29 : Simulation 3D d'une mission sur Mars. ---- 50

Figure 1.30 : L'utilisation de l'infographie dans le domaine de la technologie de l'information. ---- 50

Figure 1.31 : Utilisation de l'infographie dans les médias. ---- 50

Figure 2.1 : Exemple de structures géométriques pouvons décrire un objet 3D. ---- 54

Figure 2.2 : Des objets 3D rendus avec différentes techniques de placage de texture et différentes techniques d'ombrage. ---- 54
Figure 2.3 : L'effets de la couleur sur l'œil humain (a) et sur l'apparences des scènes 3D (b). ---- 56
Figure 2.4 : Les trois types de lumière les plus utilisées en infographie 3D.(a) Lumière directionnelle. (b) Lumière ponctuelle. (c) Lumière volumétrique. ---- 56
Figure 2.5 : Scènes 3D rendues avec une lumière directionnelle. ---- 57
Figure 2.6 : Exemples de scènes 3D rendues avec des lumières ponctuelles. ---- 57
Figure 2.7 : Exemples de scènes 3D rendues avec des lumières volumétriques. ---- 57
Figure 2.8 : Différence entre l'illumination locale et l'illumination globale. (a) Illumination locale. (b) Illumination globale. ---- 58
Figure 2.9 : Les trois principaux composant de l'ombrage. (a) Ombrage ambiant. (b) Ombrage diffus. (c) Ombrage spéculaire. (d) Combinaison des trois composants. ---- 60
Figure 2.10 : Cas de figures pris en charge par les modèles d'ombrage. ---- 60
Figure 2.11 : Différents modèles d'ombrage. (a) Modèle de Lambert. (b) Modèle de Gouraud. (c) Modèle de Phong. ---- 62
Figure 2.12 : Différence entre le modèle d'ombrage de Phong et de Blinn-Phong. ---- 64
Figure 2.13 : Principe du placage de texture. (a) Espace objet. (b) Surface S dans l'espace objet. (c) Espace texture. ---- 64
Figure 2.14 : Exemple d'ombrage en utilisant différentes cartes. (a) Carte diffuse. (b) Carte d'éclairage. (c) Diffuse × Eclairage. ---- 65
Figure 2.15 : Exemple d'ombrage. (a) Utilisation d'une carte diffuse seulement. (b) Combinaison entre la carte diffuse et la carte spéculaire. ---- 65
Figure 2.16 : Comparaison entre le placage de texture et le placage de bosselures. (a) Placage de textures. (b) Carte de hauteurs. (b) Placage de bosselures. ---- 66
Figure 2.17 : Principe du placage de bosselures. (a) Carte diffuse. (b) Carte de hauteurs. (c) Perturbation des normales selon la carte de hauteurs H. ---- 66
Figure 2.18 : Comparaison de rendu entre le placage de texture (a) et le placage de bosselures (b). ---- 67
Figure 2.19 : Les éléments fondamentales du placage de normales. (a) Carte de hauteurs. (b) Carte de normales. (c) Espace tangent. ---- 67
Figure 2.20 : Principe du placage de déplacement par sommet. ---- 69
Figure 2.21 : Placage de déplacement par sommet appliqué sur différents objets 3D. ---- 70
Figure 2.22 : Exemple de processus de tessellation par le GPU. ---- 70
Figure 2.23 : Différentes techniques du placage de texture [46]. (a) Placage de textures. (b) Placage de bosselures. (c) Placage de déplacement par sommet. (d) Placage de déplacement par pixel. ---- 71
Figure 2.24 : (a) Texture (carte diffuse). (b) Carte de profondeurs. (c) Carte de normales. (d) Carte de cônes. ---- 72
Figure 2.25 : Traçage de rayons dans la carte de déplacement [46]. (a) Vue 3D du traçage de rayons. (b) La tranche du relief incluant le rayon de vue. (c) La carte de profondeurs. (d) La texture correspondante. -- 72
Figure 2.26 : Exemples d'objets 3D rendus avec le placage de déplacement par pixel. ---- 73
Figure 3.1 : Différence entre le placage de normales (a) et le placage de parallaxe (b). ---- 77
Figure 3.2 : Différentes méthodes du placage de parallaxe pour la recherche du point d'intersection [48]. ---- 78
Figure 3.3 : Comparaison entre le placage de parallaxe simple (a) et le placage de parallaxe itératif (b) [51]. ---- 80
Figure 3.4 : Les deux étapes de la recherche du point d'intersection utilisées dans le placage de relief [48]. (a) Recherche linéaire. (b) Recherche binaire. ---- 81
Figure 3.5 : Théière rendue avec le placage de relief en utilisant différentes cartes de déplacement [59].82
Figure 3.6 : Comparaison entre la recherche binaire et la recherche sécante [58]. ---- 82

Figure 3.7 : Principe de la technique de traçage de sphères [61]. (a) La carte de profondeurs visualisée en 2D. (b) La carte de déplacement correspondante. (c) Processus de la recherche. ---- 83
Figure 3.8 : Rendu d'un texte en 3D en utilisant la technique de traçage de sphère [61]. ---- 83
Figure 3.9 : Principe de la technique du traçage de cylindres. (a) Traçage de rayons. (b) Recherche binaire [63]. ---- 83
Figure 3.10 : Objets 3D rendus avec la technique du traçage de cylindres [63]. ---- 84
Figure 3.11 : Technique de traçage de cônes. (a) Les deux versions de cône, conservative en bleu et relaxée en vert. (b) Processus de la recherche [65]. ---- 84
Figure 3.12 : Cube rendu avec la technique de traçage de cônes relaxés [65]. ---- 84
Figure 3.13 : Comparaison entre deux rendus par la technique de traçage de cônes relaxés. (a) Algorithme quadratique [65]. (b) Algorithme linéaire [46]. ---- 85
Figure 3.14 : (a) Deux dilatations différentes avec $\delta_1 < \delta_2$. (b) Zone de sécurité pour quatre dilatations. (c) Calcul de l'intersection du rayon de vue avec la zone de sécurité [68]. ---- 85
Figure 3.15 : Objets rendu à l'aide de l'encodage de l'espace vide par dilatation et érosion [68]. ---- 85
Figure 3.16 : Principe du placage de déplacement pyramidale. Un rayon est tracé hiérarchiquement dans l'ordre descendant en utilisant une structure pyramidale [70]. ---- 86
Figure 3.17 : La carte de déplacement pyramidale [69]. ---- 86
Figure 3.18 : Comparaison entre le placage de relief (a) et le placage de déplacement pyramidale (b) [70]. ---- 86
Figure 3.19 : Exemples de placage d'extrusion par pixel [74]. ---- 87
Figure 3.20 : Exemples d'objets 3D crées en entier à l'aide de l'extrusion par pixel [78]. ---- 87
Figure 3.21 : Exemples d'objets 3D crées en entier à l'aide de la révolution par pixel [78]. ---- 87
Figure 3.22 : Recherche de l 'intersection du rayon de vue avec le microrelief dans le cas d'une surface plane et dans le cas d'une surface courbée [84]. ---- 88
Figure 3.23 : Rendu d'un cylindre par deux techniques [84]. (a) Placage de relief. (b) Approximation quadratique. ---- 89
Figure 3.24 : Principe du traitement de la silhouette à base de l'espace tangent [86]. ---- 89
Figure 3.25 : Objets 3D rendus avec traitement de la silhouette à base de l'espace tangent [86]. ---- 89
Figure 3.26 : Principe du placage à l'aide de coquille [87]. (a) Vecteurs utilisés pour définir le deuxième système de coordonnées. (b) Prisme avec ses triangles résultants. (c) Traçage de rayons dans le prisme extrudé. ---- 90
Figure 3.27 : Objets rendus avec le placage à l'aide de coquille [87]. ---- 90
Figure 3.28 : Variation de déplacement en fonction de la vue et la nature de la surface [92]. (a) Surface plane. (b) Surface courbée. ---- 91
Figure 3.29 : Images VDM synthétisées à partir d'une carte de profondeurs [92]. (a) La carte de profondeurs. (b-e) Images VDM pour différentes directions de vue. ---- 91
Figure 3.30 : Objets rendus avec le placage de déplacement dépendant du point de vue [92]. ---- 91
Figure 3.31 : Auto-ombrage avec la technique du placage des ombres [59]. (a) Traçage de rayons de lumière. (b) Placage de relief avec pris en charge de l'auto-ombrage. ---- 92
Figure 3.32 : (a) Carte d'horizon. (b) Placage de bosselures. (c) Placage d'horizon [96], [97]. ---- 93
Figure 3.33 : (a) Auto-ombrage par carte de visibilité [98]. (b) Rendu basé sur une carte de visibilité. ---- 93
Figure 4.1 : Recherche du point d'intersection du rayon de vue avec le microrelief. (a) Première intersection. (b) Recherche en utilisant des pas fixes. (c) Recherche en utilisant des pas variables. ---- 96
Figure 4.2 : Placage de relief avec une échelle de profondeur égale à 0.8 et un ongle de vue rasant la surface. ---- 98
Figure 4.3 : Placage de relief avec une synchronisation entre l'échelle de profondeur et l'ombrage [99]. ---- 98
Figure 4.4 : Placage de relief à une échelle de profondeur égale à 0.6 et avec deux ongles de vue différents, sans synchronisation (en haut) et avec synchronisation (en bas). ---- 99

Figure 4.5 : Placage de relief avec un ongle de vue fixe et deux échelles de profondeur (0.8 et 1.0), sans synchronisation (en haut) et avec synchronisation (en bas). ---------- 100

Figure 4.6 : (a) Synchronisation avec l'ongle de vue à une échelle de profondeur égale à 1.0. (b) Synchronisation avec l'échelle de profondeur avec une direction de vue rasant la surface. ---------- 100

Figure 4.7 : Placage dynamique de relief en utilisant deux échelle de profondeur (0.8 et 1.0) et deux angles de vue. Sans synchronisation (en haut) et avec synchronisation (en bas). ---------- 100

Figure 4.8 : Technique de traçage de cônes. (a) Technique conservative. (b) Technique relaxée. --------- 101

Figure 4.9 : Traçage de cônes hybrides.---------- 102

Figure 4.10 : Comparaison entre les trois techniques de traçage de cônes. ---------- 102

Figure 4.11 : Cylindre rendu avec les techniques originales de traçage de cônes. La silhouette n'est pas visible sur les bords des objets 3D.---------- 103

Figure 4.12 : Problème de la silhouette. (a) Une surface plane permet d'avoir une intersection contrairement à une surface courbée. (b) Maillage polygonal. (c) Sans correction de la silhouette. (d)-(e) Avec correction de la silhouette.---------- 103

Figure 4.13 : Coupe de deux surfaces quadratiques. La distance entre le rayon de vue est la quadrique est donnée par le segment PR.---------- 104

Figure 4.14 : (a) A chaque itération on se rapproche vers la surface quadratique. (b) A chaque itération, on s'éloigne de la surface quadratique. ---------- 105

Figure 4.15 : Un rayon qui atteint la profondeur 1 a atteint le fond de la carte de profondeurs caractérisant une intersection (rayon bleu). D'autre part, un rayon qui retourne à la profondeur 0 peut être écarté en toute sécurité (rayon rouge). ---------- 106

Figure 4.16 : Les défauts sont visibles dans les parties où le rayon de vue rase la surface 3D. ---------- 106

Figure 4.17 : Rectification du rayon de vue. (a) Le point P_{i+1} est rectifié selon la valeur dut_{i+1}. (b) Le point P_{i+1} est rectifié selon la valeur $-t_{i+1}^2 d$. ---------- 108

Figure 4.18 : Comparaison d'une sphère rendue par les techniques originales de traçage de cônes et par l'utilisation de la rectification du rayon de vue. (a) Technique conservative. (b) Technique relaxée.----- 108

Figure 4.19 : Un tore rendu en utilisant la rectification du rayon de vue lors du traçage de cônes.------ 109

Figure 4.20 : Rectification des cônes. (a) La surface quadratique se rapproche du point P_{i+1}. (b) La surface quadratique s'éloigne du point P_{i+1}. ---------- 110

Figure 4.21 : Comparaison d'une sphère rendue par les techniques originales de traçage de cônes et par l'utilisation de la rectification des cônes. (a) Technique conservative. (b) Technique relaxée. ---------- 110

Figure 4.22 : Un tore rendu en utilisant l'approche de rectification des cônes. ---------- 111

Figure 4.23 : Rendu de plusieurs objets 3D en temps réel avec le même processus de rectification des cônes et avec la même texture.---------- 111

Figure 4.24 : Organigramme de l'algorithme de traçage de cônes avec prise en charge de la silhouette. 112

Figure 4.25 : Comparaison d'un cylindre rendu par les techniques originales de traçage de cônes et par l'utilisation des rectifications proposées. (a) Technique conservative. (b) Technique relaxée. ---------- 116

Figure 4.26 : Interpénétration de deux objets 3D, une sphère et un tore. (a) Technique conservative. (b) Technique relaxée. ---------- 117

Figure 4.27 : Rendu d'un tore. (a) Rectification du rayon de vue. (b) Rectification des cônes. ---------- 117

Figure 4.28 : Rendu d'un tore. (a) Technique conservative avec rectification du rayon de vue. (b) Technique conservative avec rectification des cônes. (c) Technique relaxée avec rectification du rayon de vue. (d) Technique relaxée avec rectification des cônes.---------- 118

Figure 4.29 : Rendu d'un cylindre avec deux résolutions de texture : 256×256 et 2048×2048. (a) Technique conservative. (b) Technique relaxée.---------- 118

Figure 4.30 : Un tore rendu avec la technique de traçage de cônes avec correction de la silhouette en utilisant un nombre différent d'itérations linéaires (5, 10 et 25). ---------- 119

Figure 4.31 : Différents objets 3D rendus par la technique de traçage de cônes avec correction de la silhouette et en utilisant des reliefs à haute fréquence.---------- 119

Figure 4.32 : Rendu d'un vase avec la technique de traçage de cônes avec correction de la silhouette en utilisant différentes cartes et différentes valeurs de profondeurs.---------- 119

Figure 4.33 : Distorsions dus à l'utilisation de l'approximation quadratique. ---------- 120
Figure 4.34 : Comparaison de la vitesse moyenne de rendu entre les techniques de traçage de cônes et la technique du placage de relief [84]. ---------- 120
Figure 5.1 : Placage d'extrusion par pixel. (a) Placage d'extrusion de base. (b)-(c) Correction de la silhouette. ---------- 124
Figure 5.2 : Le premier point d'intersection entre le rayon de vue avec le motif extrudé. ---------- 125
Figure 5.3 : Illustration d'une carte de forme. (a) La forme de base. (b) La carte de distance. (c) La carte de forme. ---------- 125
Figure 5.4 : Processus de traçage de rayons associé au placage d'extrusion par pixel [74]. ---------- 126
Figure 5.5 : (a) Placage d'extrusion basic. (b) Placage d'extrusion étendue. (c) Placage d'extrusion biseautée. (d) Placage d'extrusion avec chanfrein. ---------- 127
Figure 5.6 : Recherche du point d'intersection dans le cas de l'extrusion biseautée [74]. ---------- 127
Figure 5.7 : Recherche du point d'intersection dans le cas de l'extrusion avec chanfrein [74] ---------- 128
Figure 5.8 : Rectification du rayon de vue. (a) Le rayon de vue est à l'intérieur du quadrique. (b) Le rayon de vue est à l'extérieur du quadrique. ---------- 128
Figure 5.9 : Rectification de la distance. (a) La distance est augmentée afin que la surface quadratique se rapproche du point P_{i+1}. (b) La distance est diminuée afin que la surface quadratique s'éloigne du point P_{i+1}. ---------- 130
Figure 5.10 : Organigramme de l'algorithme de placage d'extrusion avec pris en charge de la silhouette. ---------- 132
Figure 5.11 : Comparaison de rendu sans et avec correction de silhouette. En haut, le placage d'extrusion basique et en bas le placage d'extrusion avec correction de la silhouette. ---------- 135
Figure 5.12 : Comparaison de rendu du placage d'extrusion sans et avec correction de la silhouette. --- 135
Figure 5.13 : Placage d'extrusion biseautée (a) et avec chanfrein (b) sans et avec correction de la silhouette. ---------- 136
Figure 5.14 : Placage d'extrusion étendue avec correction de la silhouette. ---------- 136
Figure 5.15 : Comparaison de la vitesse moyenne de rendu de l'approche par rectification du rayon de vue avec le placage d'extrusion [74] et le placage de relief [84]. ---------- 137
Figure 5.16 : Comparaison de la vitesse moyenne de rendu des deux rectifications du placage d'extrusion. ---------- 138
Figure 5.17 : Distorsions dus à l'utilisation de l'approximation quadratique. ---------- 138

LISTE DES TABLEAUX

Tableau 0.1 : Environnement logiciel. 23

Tableau 0.2 : Configuration matérielle. 23

Tableau 4.1 : Comparaison de la vitesse de rendu IPS (Images Par Seconde) entre les techniques de traçages de cônes et le placage de relief [84]. 120

Tableau 4.2 : Le nombre moyen de IPS (Images Par Seconde) de la diminution de la vitesse de rendu des deux rectifications proposées par rapport aux techniques originales de traçage de cônes. 121

Tableau 5.1 : Comparaison de la vitesse de rendu IPS (Images Par Seconde) de l'approche par rectification du rayon de vue avec le placage d'extrusion [74] et le placage de relief [84]. 137

Tableau 5.2 : Comparaison de la vitesse de rendu IPS (Images Par Seconde) entre les deux approches du placage d'extrusion avec correction de la silhouette. 137

LISTE DES ALGORITHMES

Algorithme 4.1 (tMax1) : Calcul de *tmax* dans le cas où le rayon de vue est à l'intérieur de la quadrique. ---------- 106

Algorithme 4.2 (tMax2) : Calcul de *tmax* dans le cas où le rayon de vue est à l'extérieur de la quadrique. ---------- 107

Algorithme 4.3 (RectRay1) : Traçage de cônes avec rectification du rayon de vue dans le cas où celui-ci est à l'intérieur de la quadrique.---------- 108

Algorithme 4.4 (RectRay2) : Traçage de cônes avec rectification du rayon de vue dans le cas où celui-ci est à l'extérieur de la quadrique. ---------- 108

Algorithme 4.5 (RectCone1) : Traçage de cônes avec rectification des cônes dans le cas où le rayon de vue est à l'intérieur de la quadrique. ---------- 111

Algorithme 4.6 (RectCone2) : Traçage de cônes avec rectification des cônes dans le cas où le rayon de vue est à l'extérieur de la quadrique.---------- 111

Algorithme 4.7 : Traçage de cônes avec pris en charge de la silhouette.---------- 113

Algorithme 5.1(ExtRectRay1) : Extrusion basique avec rectification du rayon de vue dans le cas où le rayon de vue est à l'intérieur de la quadrique. ---------- 129

Algorithme 5.2 (ExtRectRay2) : Extrusion basique avec rectification du rayon de vue dans le cas où le rayon de vue est à l'extérieur de la quadrique.---------- 129

Algorithme 5.3 (ExtBiseauRectRay1) : Extrusion biseautée avec rectification du rayon de vue dans le cas où le rayon de vue est à l'intérieur de la quadrique. ---------- 129

Algorithme 5.4 (ExtBiseauRectRay2) : Extrusion biseautée avec rectification du rayon de vue dans le cas où le rayon de vue est à l'extérieur de la quadrique.---------- 129

Algorithme 5.5 (ExtRectDist) : Extrusion basique avec rectification de la distance.---------- 131

Algorithme 5.6 (ExtBiseauRectDist) : Extrusion biseautée avec rectification de la distance. ---------- 131

Algorithme 5.7 : Placage d'extrusion par pixel avec pris en charge de la silhouette. ---------- 133

Introduction Generale

PRESENTATION DU SUJET

Le placage de déplacement tridimensionnel par pixel est une technique utilisée par les algorithmes informatiques de modélisation et de rendu 3D. Son objectif principal est d'ajouter des détails géométriques à une surface sans augmenter sa complexité. Elle permet de créer l'illusion de profondeur et de relief en utilisant des informations de déplacement stockées dans une texture appelée *carte de déplacement*. Chaque pixel de cette carte représente une valeur qui spécifie la grandeur avec laquelle les points correspondant sur la surface 3D doivent être déplacés. Cette technique permet de créer des détails géométriques supplémentaires qui peuvent être très fins et complexes, et produit l'illusion d'une géométrie beaucoup plus détaillée.

Le placage de déplacement tridimensionnel par pixel présente plusieurs avantages significatifs. En premier lieu, elle permet d'ajouter des détails de manière efficace, sans augmenter considérablement la complexité géométrique des objets 3D. Cela se traduit par une optimisation des performances de rendu, tout en conservant un niveau élevé de détails visuels. De plus, elle est couramment utilisée en conjonction avec d'autres techniques de rendu, telles que le placage de bosselure, pour améliorer davantage les détails visuels et l'apparence réaliste des surfaces 3D. Cette méthode est également compatible avec les techniques d'illumination, ce qui permet de créer des effets d'ombre et de lumière plus précis et plus détaillés.

Malgré les efforts qui ont été fournis, le placage de déplacement par pixel souffre encore d'une problématique fondamentale. En effet, la plupart des approches proposées dans ce sens se concentre uniquement sur les surfaces planes, mais dans le cas des surfaces courbées, les bords des objets 3D ne sont pas rendus correctement. On parle alors de problème de silhouette.

Afin d'abréger le nom de la technique, nous allons nous contenter parfois de dire simplement *placage de déplacement par pixel* au lieu de *placage de déplacement tridimensionnel par pixel*.

Dans cette introduction générale, nous allons présenter les sections suivantes :

- Le cadre scientifique dans lequel se situe cette thèse.
- Les motivations qui nous ont poussés à choisir le placage de déplacement par pixel comme thème de recherche et le but ultime de cette thèse.
- Les contributions majeures apportées par nos différents travaux de recherches.
- L'environnement logiciel et les configurations matérielles qui sont utilisés pour expérimenter nos travaux de recherche.
- L'organisation du document qui décrit brièvement le contenu et le but de chacun de ses chapitres.

CADRE DE LA THESE

Au cours de ces dernières trois décennies, il y a eu un fort engouement dans le domaine de l'informatique, notamment pour l'infographie et la modélisation des scènes naturelles, mettant en avant l'utilisation d'algorithmes sophistiqués. L'objectif principal est de créer des scènes 3D d'une réalité saisissante et de reproduire des environnements existants. Grâce à l'évolution rapide des capacités de calcul des ordinateurs et du matériel graphique, les images et les scènes de synthèse présentées dans les jeux vidéo et l'industrie cinématographique sont de plus en plus détaillées, ce qui les rend extrêmement crédibles. Ainsi, l'infographie est devenue un domaine extrêmement actif que ce soit dans le domaine de la recherche ou de développement. En conséquence, les scènes 3D présentes dans divers secteurs technologiques sont de plus en plus détaillées, ce qui améliore considérablement leur apparence et les rend plus crédibles et plus réalistes. Cette évolution est le fruit de l'acquisition de la puissance nécessaire au traitement de l'imagerie numérique par les ordinateurs, permettant ainsi aux images de synthèse de s'imposer dans presque tous les domaines de la technologie et de la communication. Grâce à leur relation privilégiée avec le cerveau humain, les images jouent souvent un rôle central, contrairement au texte, au langage ou aux modélisations abstraites.

De nos jours, l'infographie 3D est omniprésente dans notre quotidien, couvrant une grande variété de domaines très différents. L'industrie du cinéma, les jeux vidéo, la conception mécanique, l'architecture, la médecine, la publicité, la visualisation scientifique, et bien d'autres encore, font un usage intensif de l'infographie en tant qu'outil de travail, profitant des développements qu'elle a connus. Bien que chaque domaine poursuive ses propres objectifs spécifiques en infographie, deux objectifs principaux sont universels : le réalisme et la vitesse de rendu. Ces objectifs sont directement liés à la puissance des processeurs graphiques, responsables de l'affichage et de la construction des scènes 3D, ce que l'on appelle le rendu 3D.

Le rendu 3D est un processus essentiel permettant de convertir les modèles 3D en images 2D, en prenant en compte les effets environnementaux de la scène à afficher pour obtenir un résultat réaliste. Il existe deux types majeurs de rendus : le rendu précalculé et le rendu temps réel, se distinguant principalement par leur vitesse de calcul et de traitement des images. Le rendu précalculé se concentre sur la qualité visuelle des images, alors que le rendu temps réel se concentre sur leur vitesse de traitement.

L'un des défis majeurs de l'infographie a toujours été de créer des détails visuels très riches sur les surfaces 3D générées par ordinateur, surtout lorsqu'il s'agit de les rendre en temps réel dans des scènes complexes. Ces détails, qui se trouvent en dessus de la surface des objets 3D, sont appelés des microreliefs. Il est essentiel de savoir les représenter et les synthétiser afin de donner l'apparence la plus réaliste possible à un objet 3D, favorisant ainsi son intégration dans une application interactive.

Malheureusement, la représentation de ces microreliefs à l'aide d'un maillage polygonal ne satisfait pas toujours la qualité de rendu souhaitée, et cela en raison de problèmes de

performances. Les maillages polygonaux sont souvent limités en termes de résolution et de complexité, ce qui rend presque difficile la représentation précise des microreliefs. De plus, l'utilisation des formules d'ombrage pour simuler ces microreliefs peut donner des résultats visuellement pauvres, ne parvenant pas à atteindre la complexité et la richesse des détails souhaités.

Ces limitations dans la représentation des microreliefs ont longtemps été un défi pour le rendu temps réel. Les techniques de placage de texture jouent un rôle essentiel dans le domaine de l'infographie, et notamment dans l'apparence de ces microreliefs, car la recherche du réalisme reste toujours une priorité. Combiner avec d'autres techniques de rendu, les techniques de placage de texture s'avèrent jusqu'à présent les mieux adaptées pour donner l'aspect du réalisme aux scènes 3D. Les avancées dans ces techniques, telles que *les techniques de placage de bosselure* et *les techniques de placage de déplacement tridimensionnel par pixel*, ont permis de contourner ces limitations en simulant les microreliefs de manière plus réaliste, tout en préservant des performances satisfaisantes. Ces techniques permettent d'ajouter des détails subtils à la surface des objets 3D, et cela à travers des textures spéciales qui modifient leur apparence en fonction de l'angle de vue et de l'éclairage.

Dans le cadre de cette thèse, nous explorons les différentes techniques du placage de déplacement par pixel afin de générer des scènes 3D qui présentent un aspect réel et plus authentique. Dans le même contexte, notre recherche s'est focalisée sur l'amélioration significative des meilleures techniques existantes, à savoir la technique du placage de relief, la technique de traçage de cônes et la technique de placage d'extrusion. Les améliorations proposées sont portées essentiellement sur le temps de rendu, la qualité visuelle et le traitement de la silhouette.

MOTIVATIONS ET BUT

Notre recherche s'inscrit dans le domaine de l'informatique, plus précisément dans la modélisation 3D et les visites virtuelles d'édifices, avec pour objectif l'utilisation d'algorithmes pour la reconstruction en trois dimensions de monuments historiques. L'objectif ultime est de constituer une base de données du patrimoine bâti, exploitant les avancées de l'informatique pour préserver et promouvoir notre héritage culturel. Toutefois, la mise en œuvre d'un tel projet nécessite un équipement professionnel coûteux et un budget considérable. C'est pourquoi nous orientons notre choix vers la dimension théorique de ce projet, en mettant l'accent sur un problème spécifique : garantir une interactivité fluide lors des visites virtuelles de bâtiments.

Généralement, les scènes 3D sont complexes et très riches en détails. Cependant, étant donné que ces détails se génèrent à partir des maillages polygonaux, le nombre de polygones et de sommets sera si élevé que la carte graphique doit disposer d'énorme capacité de calcul. En effet, la carte graphique doit produire un nombre d'images suffisant pour que l'animation au sein des visite virtuelles soit très fluide. Pour remédier à ce problème, la solution adoptée

était de simplifier considérablement les maillages et de supprimer certains éléments de la scène. Cela a permis de résoudre le problème d'interactivité, mais au prix d'une scène visuellement moins riche et réaliste.

Notre motivation est de poursuivre les travaux de recherche déjà publiés dans ce domaine, en présentant des améliorations et de nouvelles solutions. Le but ultime est de préserver la richesse visuelle des scènes 3D complexes, tout en garantissant une interactivité fluide et une expérience réaliste lors des visites virtuelles. Nous souhaitons repousser les contraintes liées aux ressources matérielles et développer des algorithmes novateurs pour atteindre ces objectifs ambitieux.

CONTRIBUTIONS

Nos contributions sont étroitement liées à notre objectif principal, qui est l'amélioration de la technique du placage de déplacement tridimensionnel par pixel. Pour atteindre cet objectif, nous avons exploré différentes approches et développé des solutions novatrices. Nous avons cherché des moyens pour améliorer la qualité de rendu des modèles 3D sans compromettre leur apparence visuelle. Cela impliquait de trouver un équilibre délicat pour la vitesse de rendu des microreliefs au niveau des surfaces et notamment aux bords des objets 3D. Dans cette thèse, nous allons présenter quatre contributions majeures.

La première contribution porte sur la technique du placage de relief, elle propose une synchronisation combinée entre l'échelle de profondeur des microreliefs, l'ongle de vue et le nombre d'itérations de l'algorithme de traçage de rayons. Contrairement aux techniques existantes qui supposent une échelle fixe et un nombre d'itérations prédéterminé, notre approche permet d'adapter le nombre d'itérations selon une variation flexible et dynamique de l'échelle de profondeur et de l'angle de vue. En effet, la technique du placage de relief est une technique très simple et produit des résultats satisfaisants à un coup relativement faible. Cependant, dans le cas où l'échelle de profondeur est très grande ou l'angle de vue rase la surface de l'objet 3D, le nombre d'itérations doit être considérablement augmenter pour que les objets 3D soient rendus correctement. La synchronisation proposée est particulièrement utile dans des scénarios où une même carte de déplacement doit être utilisée à des échelles différentes ou lorsque la profondeur et l'angle de vue doivent être animés et constamment mises à jour.

La deuxième contribution porte sur la technique de traçage de cônes. Cette technique utilise une phase de prétraitement durant laquelle l'espace vide autour de chaque pixel de la carte de profondeurs est calculée sous forme de cônes ouverts vers le haut. L'espace calculé contribue significativement à l'accélération de l'algorithme de traçage de rayons, et par conséquent, augmente la vitesse de rendu. La technique a été présentée en deux versions, la technique conservative et la technique relaxée. Ces deux techniques se différencient par la méthode utilisée pour le calcul des cônes pendant la phase de prétraitement d'une part, et d'autre part, par l'algorithme de traçage de cônes utilisé durant la phase de rendu. Chacune de ces techniques possèdent des avantages et des inconvénients et le choix entre les deux

dépend des caractéristiques des objets 3D et les spécificités du projet. Nous présentons alors une troisième technique appelée la technique hybride, qui agit en tant qu'interpolateur entre les deux autres.

En général, les techniques de traçage de cônes produisent des résultats qui conformes aux attentes de qualité visuelle et de vitesse de rendu. Cependant, cette observation est spécifique aux surfaces planes. En présence de surfaces courbées, la silhouette n'est pas visible aux bords des objets 3D. C'est-à-dire que les microreliefs ne sont pas rendus correctement et ils coïncident avec la surface 3D. A partir de cette observation, la troisième contribution vise à surmonter cette limitation en proposant deux nouveaux algorithmes de traçage de cônes. L'amélioration proposée dans cette contribution repose sur l' approximation quadratique de la surface de l'objet 3D au niveau de chaque sommet du maillage correspondant. La surface approximative est calculée dans une étape de prétraitement, et durant la phase de redu, elle est utilisée par l'algorithme de traçage de cônes, afin de déterminer les parties de l'objet 3D appartenant à la silhouette. En ce qui concerne la qualité visuelle, les deux algorithmes proposés produisent des résultats presque similaires, ce qui n'est pas le cas pour la vitesse de rendu.

La quatrième contribution porte sur la technique du placage d'extrusion. C'est une technique qui permet de produire des motifs extrudés sur des objets 3D, en exploitant les principes du placage de déplacement par pixel. Quatre types d'extrusion ont été présentés, l'extrusion basique, l'extrusion étendue, l'extrusion biseautée et l'extrusion avec chanfrein. Cette technique souffre de la même limitation que celle évoquée dans la troisième contribution, à savoir la pris en charge de la silhouette. Alors pour chaque type d'extrusion, nous avons proposée deux nouveaux algorithmes de placage d'extrusion par pixel en se basant sur l'approximation quadratique. La qualité visuelle des objets rendus par les algorithmes proposés est presque similaire. Mais concernant la vitesse de rendu, la différence est obtenue seulement au niveau de l'extrusion basique.

Généralement, les contributions proposées ont été axées sur la recherche de solutions permettant de maintenir la richesse visuelle des scènes 3D complexes, tout en garantissant une vitesse de rendu optimale et une interactivité fluide.

ENVIRONNEMENT LOGICIEL ET CONFIGURATION MATERIELLE

L'environnement logiciel et la configuration matérielle sont des aspects essentiels pour assurer des performances optimales et une expérience utilisateur fluide dans divers domaines, notamment dans le domaine de l'infographie 3D. L'environnement logiciel comprend à la fois le système d'exploitation utilisé, tel que Windows, MacOs ou Linux, ainsi que les logiciels et les pilotes nécessaires pour prendre en charge les fonctionnalités graphiques avancées. La configuration matérielle, quant à elle, concerne les composants physiques de l'ordinateur, tels que le processeur (CPU), la mémoire (RAM), la carte

graphique (GPU), le stockage, etc. La puissance et les spécifications de ces composants jouent un rôle crucial dans les performances globales du système.

En générale, pour l'infographie 3D, il est recommandé d'avoir une configuration matérielle robuste comprenant un processeur puissant, une quantité suffisante de mémoire vive, une carte graphique haut de gamme et un stockage rapide. Un environnement logiciel bien optimisé et à jour, ainsi que la compatibilité matérielle avec les logiciels utilisés, sont également essentiels. Dans le cadre de notre thèse, et pour expérimenter nos travaux de recherches, nous avons utilisé un même environnement logiciel (Tableau 0.1) et deux configurations matérielles différentes que nous allons les différencier par ***Config.1*** et ***Config.2*** (Tableau 0.2).

Tableau 0.1 : Environnement logiciel.

Système d'exploitation	IDE	Langage de programmation	Bibliothèque de programmation 3D
Windows 10	Microsoft Visual Studio	C++	OpenGL/GLSL 4.6

Tableau 0.2 : Configuration matérielle.

Configuration	CPU	RAM	Stockage	GPU	Mémoire dédiée
Config.1	Intel Core-i7-4510U-2GH-4CPUs	8Go	HDD	GeForce-GT-840M	4Go
Config.2	Intel Core-i7-7820HQ-2.9GH-8CPUs	16Go	SSD	Quadro M2200	

ORGANISATION DU DOCUMENT

Suite à une analyse préliminaire visant à découvrir et comprendre les objectifs du projet, nous avons défini notre approche de mise en œuvre et structuré ce document de la manière suivante :

Dans le premier chapitre, nous abordons les fondements de base de l'infographie 3D. Nous introduisons une brève historique de l'infographie en explorant les différentes périodes clés de son développement et qui ont contribué à son avancement. Juste après, nous présentons les différentes étapes de la création des scènes 3D et les différents types de rendu. Ensuite, nous introduisons le pipeline graphique qui joue un rôle essentiel dans la réalisation de l'infographie 3D, en décrivant ses étapes principales au niveau des cartes graphiques modernes. A la fin du chapitre, nous citons quelques domaines d'applications de l'infographie, allant du divertissement à l'architecture, en passant par la médecine et l'ingénierie, offrant ainsi de vastes possibilités de visualisation et de communication de concepts complexes.

Dans le deuxième chapitre, nous introduisons les principes fondamentaux des techniques d'ombrage et de placage de textures. Nous commençons tout d'abord par décrire la structure globale d'un objet 3D, constituée de sommets et de polygones, formant ainsi sa base essentielle. Nous introduisons par la suite le concept de l'illumination qui joue un rôle primordial dans l'apparence des scènes 3D, et qui prend en considération plusieurs facteurs clés tels que l'éclairage de la scène et la nature des matériaux des objets 3D. Les deux

dernières sections du chapitre sont consacrées aux modèles d'ombrage et aux techniques de placage de textures en explorant ses différentes variantes.

Le troisième chapitre se penchera sur l'état de l'art du placage de déplacement tridimensionnel par pixel. Nous examinerons en détail les différentes solutions proposées pour générer les microreliefs sur les objets 3D à partir de la carte de profondeurs. Nous discuterons également des techniques permettant de prendre en charge l'auto-ombrage et le traitement de la silhouette.

Le quatrième chapitre sera consacré à nos contributions concernant la technique de placage de relief et la technique de traçage de cônes. Dans la première partie de ce chapitre, nous présentons le placage dynamique de relief, qui consiste à synchroniser le nombre d'itérations de l'algorithme de traçage de rayons avec l'échelle de profondeur et l'angle de vue. Dans la deuxième partie, nous proposons la version hybride de la technique de traçage de cônes. Et dans la troisième partie, nous présentons les solutions proposées concernant le traitement de la silhouette pour la technique de traçage de cônes.

Le cinquième chapitre sera consacré au placage d'extrusion par pixel. Nous commençons par décrire brièvement les concepts de la technique et ses différents types. Ensuite, nous présentons les solutions proposées pour résoudre le problème de la silhouette.

Ce mémoire se termine par une conclusion générale qui résume en détail nos travaux de recherche. Nous mettons en évidence les principales contributions en soulignant les résultats obtenus et les implications importantes de notre recherche. De plus, nous identifions les perspectives futures et les différentes pistes de recherche qui pourraient être explorées pour améliorer encore davantage les solutions proposées. Nous soulignons l'importance de ces perspectives pour étendre et approfondir notre compréhension du sujet, ainsi que pour répondre aux défis et aux besoins actuels du domaine. Cette conclusion générale constitue une synthèse de notre travail de recherche, et ouvre la voie à de nouvelles possibilités d'exploration et d'amélioration dans les domaines abordés dans cette thèse.

Chapitre 1

Fondements de base de l'infographie 3D

1.1 INTRODUCTION

L'infographie est l'art de représenter des informations sous forme graphique pour une présentation visuelle claire et attractive. Il s'agit d'un ensemble d'algorithmes et de techniques informatiques pour la manipulation et la représentation des données sous formes d'images. En utilisant l'infographie, les informations peuvent être transmises de manière plus rapide et plus facile à comprendre que les textes traditionnels, ce qui en fait un outil très utile pour les présentations, les rapports et les publications.

L'objectif de l'infographie est la création ou le traitement d'images à l'aide d'ordinateurs. Ces images peuvent être en deux ou en trois dimensions et sont utilisables sur n'importe quel support. La création ou le traitement d'images peut se faire soit en temps réel, comme c'est le cas pour les applications interactives telles que la simulation 3D, les visites virtuelles et les jeux vidéo, où la vitesse est un facteur déterminant. Soit en post-production, pour des domaines n'impliquant pas d'interaction avec l'utilisateur, tels que le cinéma, la publicité, la modélisation scientifique, où la qualité est un facteur plus important. Dans les deux cas, on parle alors de rendu.

Contrairement à l'infographie 2D qui opère sur des plans à deux dimensions, l'infographie 3D repose sur les techniques de modélisation 3D pour créer et représenter les données et les informations en 3 dimensions et de manière plus réaliste. Cependant, la création des images 3D peut être plus complexe et plus coûteuse que la création des images 2D en raison des technologies utilisées et des compétences requises.

Vu l'importance de l'image dans plusieurs domaines scientifiques, dans lesquels des informations complexes et importantes sont manipulées et en raison de la demande élevée des consommateurs en matière de qualité et d'optimisation de traitement, l'infographie ne cesse toujours de pousser les ingénieurs et les chercheurs à satisfaire ses besoins et ses lourdes exigences que ce soit au niveau matériel ou au niveau logiciel. En outre, l'infographie moderne reposent dans sa création sur les technologies avancées du matériel informatique, notamment les unités de traitement parallèles dédiées spécialement pour le traitement des données graphiques, ces unités sont tout simplement des puces électroniques regroupées dans des cartes d'extensions appelées cartes graphiques ou GPU.

L'objectif de ce chapitre est de décrire d'une manière générale les procédures et les éléments qui contribues à la création des scènes 3D. Il expose les sections suivantes :

- Une brève historique sur l'infographie qui décrit les périodes les plus importantes dans son histoire et comment elle a été évoluée au fils des ans.
- Une description générale des étapes de la création des scènes 3D et les éléments qui entre en jeux tout en mettant l'accent sur les différences types de rendu.
- Une description du pipeline graphique qui décrit comment les étapes de la création de des scènes 3D sont implémentées au niveau des cartes graphiques modernes.
- Les différents domaines de l'application de l'infographie 3D.

1.2 HISTORIQUE

L'histoire de l'infographie remonte à plusieurs siècles, avec des illustrations et des graphiques qui ont été utilisés pour représenter des informations et communiquer des idées. Cependant, c'est au XIXe siècle que l'utilisation de la statistique a véritablement permis l'émergence de l'infographie moderne. En effet, les scientifiques et les statisticiens ont commencé à utiliser des graphiques pour représenter des données numériques, ce qui a permis une meilleure compréhension de ces derniers.

L'arrivée de l'informatique au XXe siècle a considérablement accéléré le développement de l'infographie. Les premiers ordinateurs, qui étaient principalement utilisés pour des calculs complexes, ont également permis la création de graphiques en 2D et ont largement contribués à l'évolution de l'infographie.

L'histoire de l'infographie est un parcours passionnant qui a commencé par l'utilisation de simples illustrations pour représenter des informations jusqu'à l'utilisation des technologies de pointe pour créer des infographies 3D plus complexes. L'infographie a évolué au fil des ans pour devenir un moyen puissant pour communiquer l'information de manière visuelle et attrayante. L'histoire de l'infographie peut être divisée en quatre périodes qui sont les années 1920-1960, les années 1960-1980, les années 1980-2000 et finalement les années 2000 a aujourd'hui.

1.2.1 Les années 1920-1960

C'est au début des années 1920 que l'infographie a commencé son développement et son évolution grâce à l'avènement de nouvelles technologies et à l'essor de la communication visuelle. En effet, l'introduction de la gravure sur métal a permis la création des illustrations plus détaillées et plus fines, ainsi que la production en série de revues et de livres illustrés. Cela a contribué à la popularité de l'infographie en tant que moyen de transmettre des informations complexes de manière attrayante et accessible.

L'évolution de la photographie en couleur et de la typographie dans les années 1940-1950 a aussi stimulé le développement de l'infographie pour les journaux, les magazines et les livres pour enfants qui utilisent différents types de graphiques pour illustrer des sujets complexes tels que les cartes politiques, les diagrammes financiers, les statistiques, les dessins de personnages et de scènes, etc.

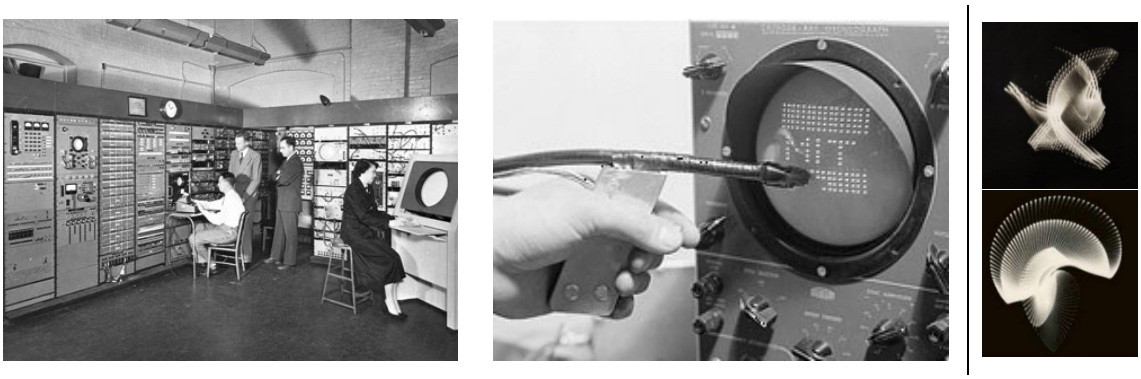
La demande élevée de l'infographie dans plusieurs secteurs a poussé les chercheurs à réfléchir concrètement à l'informatisation des processus liés à la création des graphiques. Cela s'est concrétisé au début des années 1950 où le premier ordinateur à affichage graphique a été développé au sein de l'Institut de Technologie de Massachusetts (MIT) [1], il s'agit de *Whirlwind I* (Figure 1.1a).

Cet ordinateur utilise un écran à tube cathodique et un stylo lumineux comme périphérique d'entrée afin de permettre à l'utilisateur de pointer vers des objets affichés ou de dessiner sur l'écran. C'était parmi les premiers ordinateurs électroniques numériques qui

fonctionnaient en temps réel. Dans cette même période, les premières images graphiques ont été créées par Ben Laposky via l'utilisation d'un oscilloscope pour générer une illustration de forme d'onde (Figure 1.1b).

Ces deux évènements majeurs ont donné naissance au terme *"Computer Graphics"* au 1965 qui signifie Infographie [2]. Et à partir de cette année, l'infographie n'a jamais cessé d'évoluer pour répondre aux différents besoins dans différents domaines que ce soit pour la vie quotidienne ou pour les domaines scientifiques.

Les années 1920 à 1960 ont été une période clé dans l'histoire de l'infographie, car de nouvelles technologies ont permis la création de graphiques plus détaillés et plus complexes, ainsi que l'utilisation de l'infographie dans de nouveaux médias et domaines. Cette période a également posé les fondements pour le développement futur de l'infographie informatique.



(a) (b)

Figure 1.1 : (a) Premier ordinateur à affichage graphique Whirlwind I. (b) Les premières images de Ben Laposky.

1.2.2 Les années 1960-1980

Les années 1960-1980 marquent une révolution de l'infographie où les premiers jeux vidéo et applications graphiques ont été développés grâce à l'arrivée de la technologie informatique qui a donné un coup d'avance aux capacités de l'infographie. Les ordinateurs ont facilité la production des graphiques plus complexes et plus précis, ainsi que la production des animations. Les premiers logiciels de CAO (Conception Assistée par Ordinateur) tels que *Sketchpad* et *AutoCAD* ont permis aux graphistes de concevoir des illustrations de manière plus efficace. Les premières formes de PAO (Publication Assistée par Ordinateur) ont été également développées, permettant la production de publications plus riches en illustrations.

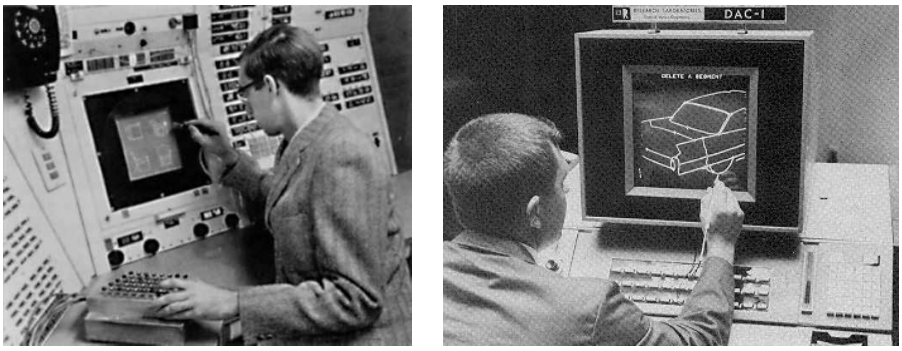
En 1962, Steve Russell développe le tout premier jeu vidéo *Spacewar* (Figure 1.2a) au sein de l'Institut de Technologie de Massachusetts [3], il s'agit d'un jeu de combat spatial pour le mini-ordinateur PDP-1 (Figure 1.2b). Mais juste en 1966, Ralph Baer prend la relève pour être considéré par la suite come le père des jeux vidéo à cause de son invention de la console de jeux vidéo domestique *Odyssey* [4] (Figure 1.2c) et ces contributions pour le développement des jeux vidéo. La console de jeux se composait d'une unité de contrôle principale contenant tout l'équipement électronique, de deux unités de contrôle qui dirigeaient les joueurs sur l'écran et d'un ensemble de cartes de programme électroniques.

En 1963, les premiers systèmes de conception assistée par ordinateur (CAO) commencent à apparaitre. Ivan Sutherland a proposé un système dans le cadre de sa thèse de doctorat à l'Institut de Technologie de Massachusetts [5], [6] . Il s'agit de *Sketchpad* (Figure 1.3a), l'idée est de permettre à l'utilisateur d'esquisser une pièce mécanique sur un écran en posant des contraintes sur cette même pièce et laisser l'ordinateur calculer sa conception. Dans la même année, General Motors et IBM ont développé le premier system CAO industriel [7] utilisé pour aider les ingénieurs à concevoir des voitures (Figure 1.3b).



(a) (b) (c)

Figure 1.2 : (a) Capture d'écran du jeu *Spacewar*. (b) Le mini-ordinateur DEC PDP-1. (c) Console de jeux *Odyssey*.



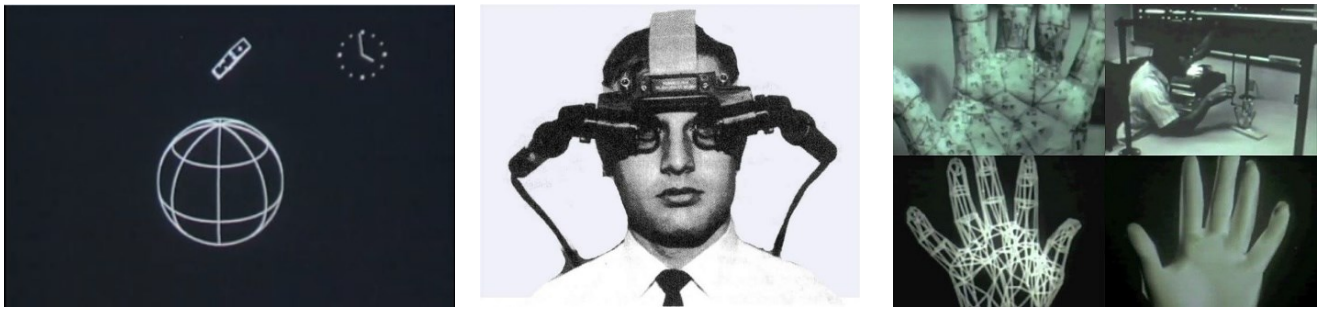
(a) (b)

Figure 1.3 : (a) Système Sketchapad. (b) Premier système CAO de General Motors et IBM.

La notion de profondeur n'était pas présente dans les images produites par les premiers systèmes de conception ou par les premiers jeux vidéo. C'est-à-dire toutes les scènes et les dessins sont créés en prenant en compte deux dimensions seulement, en parle alors de l'infographie 2D ou toutes les opérations liées aux calculs mathématiques sont effectuées sur un plan à deux dimensions.

L'introduction de la troisième dimension a donnée naissance à l'infographie 3D suite à la réalisation du premier film d'animation 3D par le laboratoire Bell Telephone en 1963. Le film dur 1.25min et montre le mouvement des satellites et leur changement d'altitude autour de l'orbite de la terre (Figure 1.4a). De l'animation 3D à la réalité virtuelle, Ivan Sutherland a créé le premier système d'affichage de réalité virtuelle en 1968 [8] qui contenait deux images filaires distinctes et donnait un effet 3D (Figure 1.4b).

Les années 1970 marquent un changement majeur dans l'infographie grâce à l'apparition d'un film produit par Edwin Catmull [9] montrant une animation 3D crée par ordinateur de sa propre main (Figure 1.4c). Le film est considéré comme une étape importante dans l'histoire de l'infographie car lors de sa création, Catmull aborde des concepts qui deviendront par la suite des principes fondamentaux de l'infographie.



(a) (b) (c)

Figure 1.4 : (a) Capture d'écran du premier film d'animation 3D. (b) Premier système de réalité virtuelle. (c) Animation 3D créé par ordinateur [9].

Une autre avancée majeure dans l'infographie a été créée en 1976 pour dessiner une représentation 3D d'un objet sur l'écran, et qui a servi de base à la plupart des développements futurs de l'infographie. Il s'agit d'un model 3D d'une théière ordinaire crée par Martin Newell [10] afin de tester ses algorithmes permettant un affichage réaliste des formes 3D (Figure 1.5). Ce modèle est devenu l'objet de référence le plus célèbre au monde au sein de la communauté de l'infographie 3D.

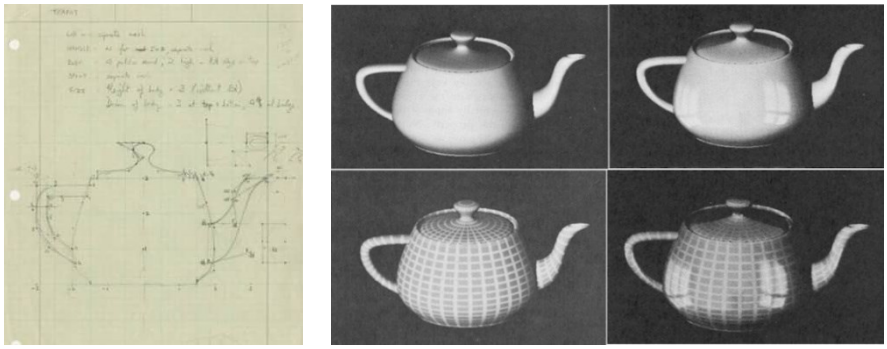


Figure 1.5 : Modélisation 3D de la théière la plus célèbre au mode de l'infographie 3D [10].

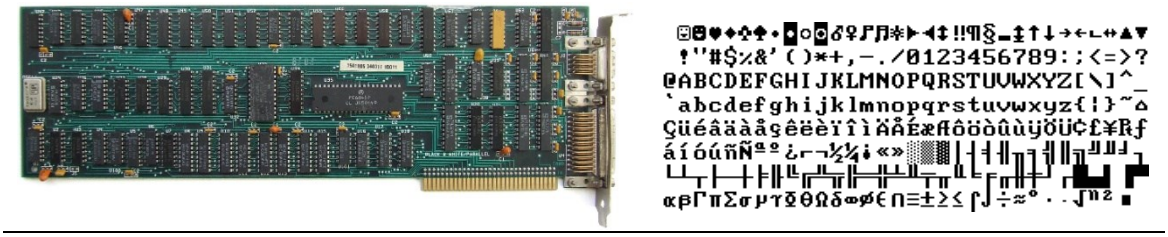
1.2.3 Les années 1980-2000

L'avènement de la CAO (Conception Assistée par Ordinateur) et de la PAO (Publication Assistée par Ordinateur) a permis aux graphistes de produire des illustrations complexes avec une plus grande rapidité et une plus grande précision. Les graphistes ont commencé à utiliser des techniques telles que la superposition et la transparence pour créer des illustrations plus riches en textures et en détails. Les techniques de la 3D sont également apparues, mais n'ont pas été largement utilisées dans l'infographie jusqu'à la fin du XXe siècle. Avec de tels développements, la modernisation et la commercialisation du matériel destiné purement au traitement graphique ont commencé dans les années 1980. De nombreux microprocesseurs et microcontrôleurs ont été développés pour donner naissance à de nombreuses cartes graphiques. Mais avant qu'il y ait la carte graphique que nous la connaissons aujourd'hui, il n'y avait plus que la carte d'affichage vidéo. Le développement des cartes d'affichage vidéo a commencé par IBM en 1981 lorsqu'ils ont introduit une carte d'affichage monochrome (Figure 1.6a) appelée MDA (Monochrome Display Adapter) pour l'utiliser dans ses ordinateurs. Cette carte permet uniquement l'affichage du texte est des symboles mais à haute résolution. En intégrant ce mode d'affichage monochrome, la carte MAD sera remplacée par la carte HGC (Hercules Graphics Card) introduit par Van

Suwannukul au sein de son entreprise Hercules Computer Technology en 1982 et qui dispose davantage le mode graphique (Figure 1.6b).

En profitant de l'invention des cartes vidéo et son développement, les années 1980 ont été aussi appelées l'âge d'or des jeux vidéo. Grace à de nombreuses entreprises de jeux vidéo comme Atari, Nintendo et Sega, l'infographie a pu être présentée au public à travers une nouvelle interface.

(a)



(b)



Figure 1.6 : (a) Carte d'affichage vidéo MDA. (b) Carte d'affichage vidéo HGC.

Les années 1990 ont été une période où l'infographie 3D a prospéré et s'est développée à grande échelle. Vue que le développement des jeux vidéo est des applications graphiques été trop pénible, la société Silicon Graphics a introduit en 1992 la première interface de programmation d'applications graphiques (API), appelée OpenGL, dans l'objectif de simplifier la programmation 3D et d'unifier ses étapes. Pour le même objectif, Direct3D a été introduite par Microsoft en 1996. Ces API font abstraction du matériel sous-jacent et empêchent le programmeur d'écrire directement du code pour manipuler les accélérateurs matériels graphiques. Au cours de cette période, de nombreuses entreprises ont commencé à développer des cartes vidéo avec plus de résolution et qui ont contribué à la révolution de l'infographie où les modèles 3D sont devenus populaires dans les jeux vidéo, le multimédia et les animations. D'ici-là, le terme GPU (Graphical Processing Unit) est né par l'introduction de la première carte graphique RIVA 128 de NVIDIA en 1997 (Figure 1.7a). Cette carte a été l'une des premières unités de traitement graphique grand public à intégrer l'accélération 3D en plus de l'accélération 2D et vidéo traditionnelle. La technologie RIVA a été ameliorée par GeForce 256 en 1999 (Figure 1.7b) en donnant un coup d'avance pour les performaces du traitement graphique notamment les jeux video 3D.



(a) (b)

Figure 1.7 : (a) Carte graphique RIVA 128. (b) Carte graphique GeForce 256.

1.2.4 Les années 2000 à aujourd'hui

L'avènement de la 3D et de l'animation a permis de nouveaux développements dans l'infographie, notamment dans les films d'animation, les jeux vidéo et les productions de réalité virtuelle. Les outils de conception en ligne et les technologies de reconnaissance de formes ont également permis une plus grande accessibilité à l'infographie pour un public plus large. Jusque-là, les développeurs des applications graphiques n'ont aucun contrôle sur les cartes graphiques, car tout est programmé par les constructeurs. Ainsi, toute la programmation et tous les calculs se font au niveau du CPU et les GPU se charges de recevoir les données et de les traiter d'une façon accélérée afin de produire les résultats sous forme d'image sur l'écran. Dans ce mode de fonctionnement, une partie de la mémoire vive (RAM) de l'ordinateur est partagée entre le CPU et le GPU afin d'assurer leur communication, de plus, cette communication consomme beaucoup de ressources matérielles et ralentie la vitesse de traitement au niveau des applications 3D.

Pour pallier à ce problème, les cartes graphiques programmables ont vu le jour au début des années 2000 (Figure 1.8). Ces cartes possèdent leur propre mémoire vive appelée VRAM (Video Random Access Memory) qui leurs permettent de stocker les données 3D et de les traiter directement par le processeur graphique indépendamment du CPU. Il existe différents types de cartes graphiques, certaines avec une programmation limitée et d'autres avec une programmation avancée.

(a)



GeForce 3



Radeon 8500

(b)



GeForce FX



Radeon 9700

Figure 1.8 : Les premières cartes graphiques programmables. (a) Programmation limitée. (b) Programmation avancée.

Les cartes graphiques avec une programmation limitée sont généralement plus simples et moins coûteuse et elles ont des capacités de calcul et des niveaux de programmation plus limitées (Figure 1.8a). Néanmoins, ce type de cartes peut être suffisant pour les utilisateurs ayant des besoins de performance graphique relativement simples tels que l'affichage de la vidéo et les jeux peu exigeants. Les cartes graphiques avec une programmation avancée sont

conçues pour traiter des tâches graphiques plus complexes et offrir aux développeurs plus de contrôle et plus de flexibilité durant le développement des applications (Figure 1.8b). Elles permettent aussi de déterminer la manière avec laquelle les applications graphiques vont être exécutées et cela à travers des langages et de programmation, des API et des outils avancés.

Le développement et l'évolution des cartes graphiques programmables durant les années 2000 ont largement contribué à l'amélioration de la qualité des images rendues par les applications graphiques jusqu'à ce qu'elles deviennent proches du monde réel. Dans cette même période et à l'aide des cartes graphiques plus modernes, le développement de la réalité virtuelle et de la réalité augmentée a donné une nouvelle impulsion à l'infographie 3D, en permettant aux utilisateurs de visualiser les données de manière interactive et plus attractive. La réalité virtuelle a permis aux utilisateurs de se déplacer dans des mondes virtuels et de voir les données en 3D de manière plus naturelle, tandis que la réalité augmentée a permis de superposer des informations en 3D sur le monde réel. La nouvelle génération des cartes graphiques permet aux développeurs d'écrire du code pour les utiliser dans des calculs spécifiques en plus de leurs tâches graphiques habituelle. Cette option a rendu les GPU populaires pour des applications telles que la simulation, le traitement de la vidéo, la reconnaissance de la voix, la reconnaissance de l'image, l'intelligence artificielle, le calcul scientifique et la cryptographie, etc.

L'histoire de l'infographie est riche et passionnante. Les outils et les technologies ont considérablement évolué pour permettre la création des graphiques plus complexes et plus attrayants, ce qui a contribué à leur popularité croissante dans la communication de l'information. Ces dates et ces évènements importants ainsi mentionnés marquent les débuts révolutionnaires des différentes branches de l'infographie en partant du matériel vers le logiciel. Depuis lors, et au cours de ces dernières années, l'infographie a connu un immense progrès grâce à l'évolution exceptionnelle du matériel informatique et graphique qui ne cesse de s'accroitre pour fournir les structures et les architectures les plus modernes afin de répondre aux exigences les plus élevés liées à la vitesse de traitement et à la qualité de rendu.

1.3 CREATION DES SCENES 3D

Aujourd'hui, l'infographie utilise des techniques de modélisation et de rendu 3D pour représenter les données et les informations de manière visuelle et interactive, ce qui peut aider à les comprendre et à les mémoriser plus facilement. Cependant, l'infographie comme discipline peut être classifié en deux axes, l'infographie 2D et l'infographie 3D. Ces deux axes s'utilisent conjointement dans biens des situations diverses. L'infographie 2D utilise des techniques telles que la mise en forme typographique, les couleurs et la superposition pour représenter des informations et des illustrations en 2 dimensions. L'infographie 3D, quant à elle, ajoute une dimension supplémentaire en utilisant des techniques telles que la modélisation, le rendu et l'animation pour représenter des informations dans un environnement virtuel en 3 dimensions (voir *Annex A*).

La création des scènes 3D implique la conception d'objets en trois dimensions ainsi que la mise en place de ces objets dans un environnement tridimensionnel. Quand en parle de scènes 3D en parle aussi de plusieurs traitements graphiques tel que l'éclairage, les effets visuels, l'animation, l'aspect du réalisme, etc. La création des scènes 3D passe par plusieurs étapes qu'on peut les résumer en deux étapes principales :

- Modélisation : qui regroupe la, création des objets 3D, l'application des matériaux, la mise en scène, l'animation, l'illumination, etc.
- Rendu : combiner les éléments de la scène pour produire une image ou une séquence vidéo cohérente.

1.3.1 Conception est modélisation

La modélisation 3D est le processus informatique de conception et de formation de l'objet 3D à l'aide d'algorithmes spécifiques, en utilisant des entités géométriques de base comme les sommets, les triangles, les polygones, les courbes, les surfaces, etc. Cette modélisation est effectuée en utilisant des logiciels de modélisation (Figure 1.9) comme par exemples Blender, Maya, 3DSMax, etc. L'objet ainsi construit est formé d'un ensemble de point appelés sommets qui forment eux-mêmes des polygones. Un polygone est une zone formée au moins de trois sommets (un triangle). L'ensemble des sommets et des polygones qui construisent l'objet 3D est appelé maillage polygonal (Figure 1.10).

Les objets 3D complexes peuvent être aussi produits et créés d'une manière procédurale en utilisant des algorithmes mathématiques et informatiques bien précis, ou à partir d'un objet très simple (plan, courbe, cube, cylindre, sphère, cône, tore, etc.) en lui appliquant des opérations telles que la subdivision du maillage, le déplacement des sommet, l'extrusion, la révolution, la déformation, les techniques fractales, etc. (Figure 1.11). Ils peuvent aussi être créés à l'aide des techniques d'acquisition des objets du monde réel comme les techniques de balayage laser et les techniques photographiques.

Une fois l'objet 3D est créé, il doit être habillé par un ensemble de couleurs qui peuvent être définies par le concepteur de l'objet ou être lues à partir des images appelées des textures créées en utilisant les différentes techniques de traitement d'images (Figure 1.12). Ces textures permettent de donner aux objets 3D des couleurs, des détails et des effets pour améliorer l'apparence des modèles, on parle alors de placage de texture. Généralement, dans cette étape on détermine uniquement la manière avec laquelle un objet 3D va être colorié ou la technique de placage de texture à utiliser, l'aspect final de l'objet est calculé pendant la phase de rendu.

La mise en scène consiste à placer les objets dans une scène 3D en appliquant des transformations matricielles (translation, rotation, mise à l'échelle, projection) afin de déterminer la taille et l'emplacement de chaque objet dans la scène (Figure 1.13). Le positionnement d'un objet dans la scène s'effectue par rapport à la caméra. La caméra est un objet fictif dans la scène 3D qui représente généralement l'œil de la personne qui regarde la scène (Figure 1.14), elle se caractérise par sa position et son angle de vue. La caméra est appelée parfois point de vue.

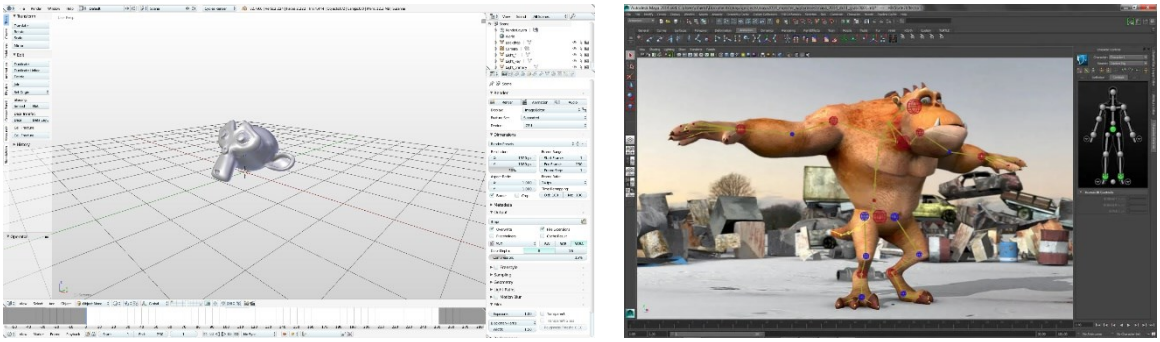


Figure 1.9 : Logiciels de modélisation et de rendu 3D. Blender (à gauche) et Autodesk Maya (à droite).

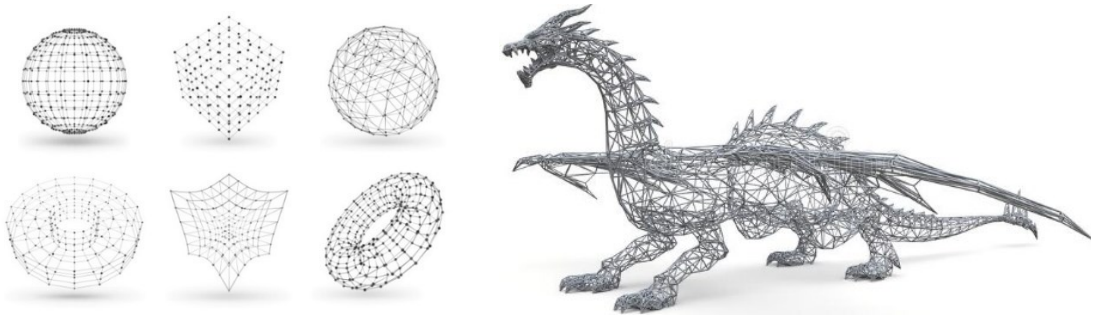
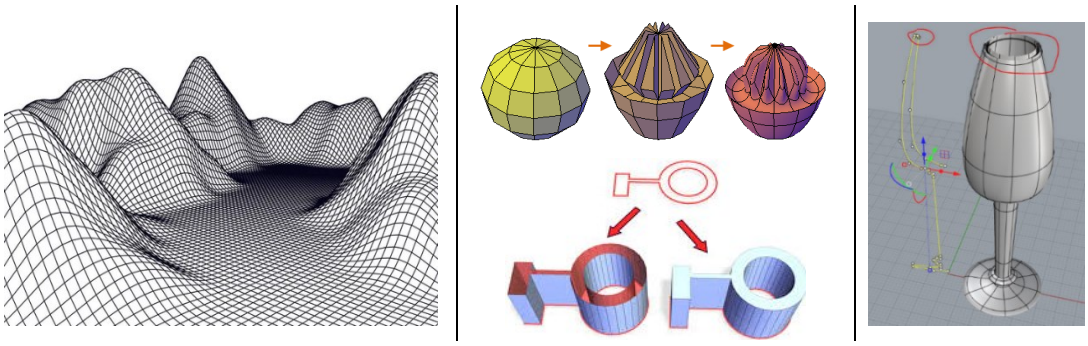


Figure 1.10 : Exemples de maillages polygonaux construits en utilisant des sommets, des triangles, des polygones, des courbes, des surfaces, etc.



(a) (b) (c)

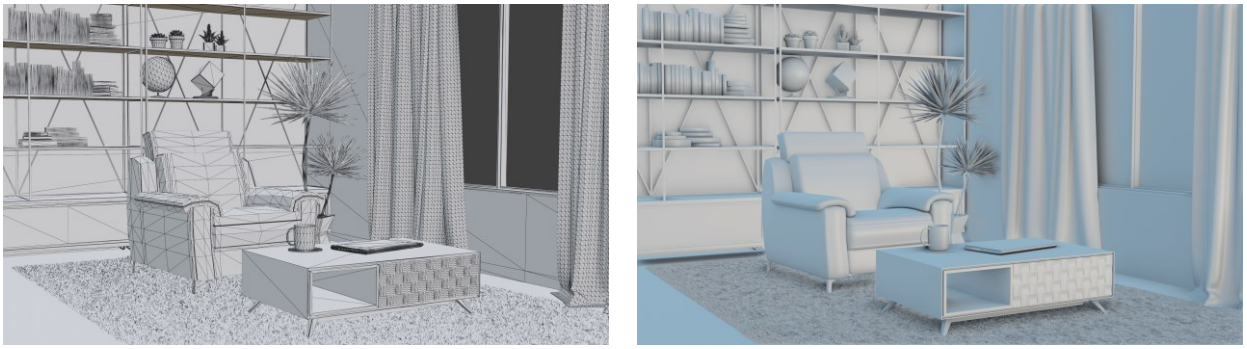
Figure 1.11 : Objets 3D créés à l'aide de différentes techniques. (a) Déplacement des sommets. (b) Extrusion des surfaces. (c) Révolution d'une courbe.



(a) (b)

Figure 1.12 : Application des textures et des couleurs a une sphère. (a) Placage de texture. (b) Application des matériaux.

L'animation est créée en affichant une série d'images similaires les unes aux autres sur l'écran de manière séquentielle pour créer l'illusion de mouvement (Figure 1.15). Les images peuvent être remplacées par un squelette virtuel et l'ordinateur calcule automatiquement les différences d'apparence entre les images successives. En trouve entre autres l'animation des objets et des personnages en définissant leurs actions et leurs mouvements pour les faire interagir avec leur environnement, la simulation des effets physiques tels que la fumée, les explosions, les flammes et les éclairs, l'animation de la caméra, l'animation des sources lumineuses, etc. Il faut noter que les gestes et les mouvements possibles des objets 3D sont définis individuellement pendant l'étape de la modélisation, et l'animations consiste à les utiliser afin de créer des séquences d'animation dans la scène.



(a) (b)

Figure 1.13 : Mise en scène des différents objets 3D. (a) Les objets sans utilisés directement dans la scène. (b) Les objets ont subis des opérations de prétraitement.

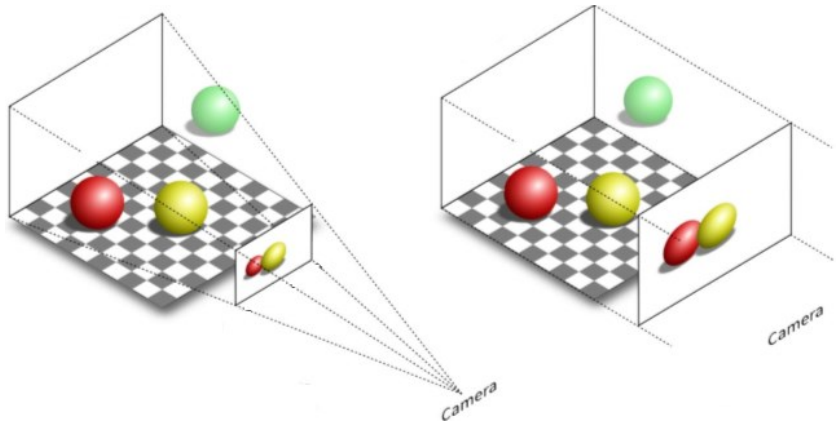
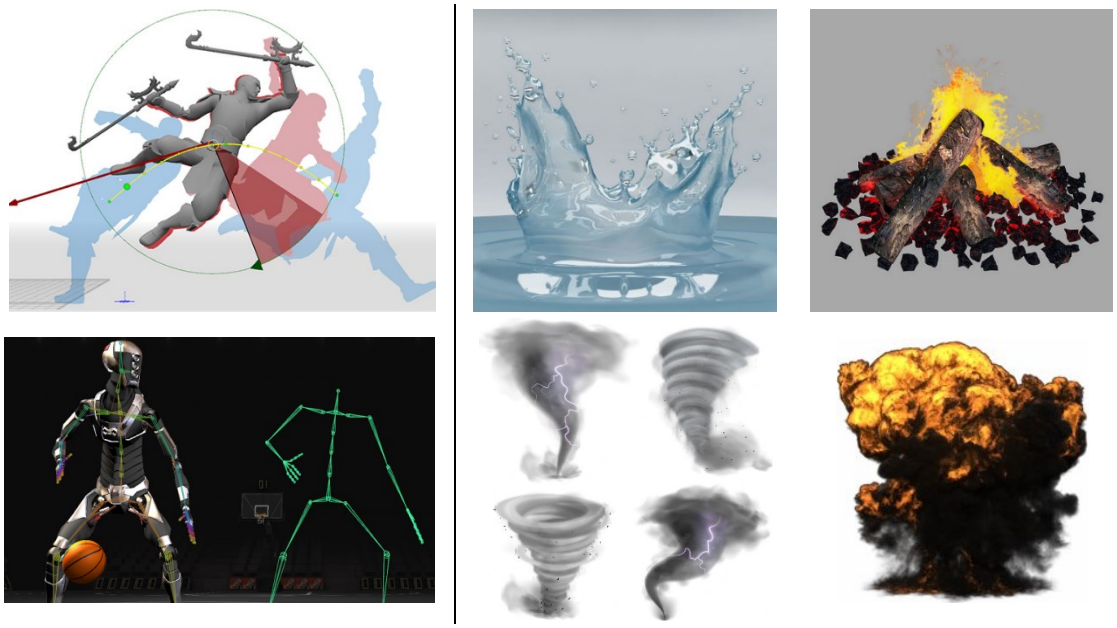


Figure 1.14 : Exemples de deux caméras représentants deux types de projections. A gauche, une caméra avec une projection perspective. A droite, une caméra avec une projection orthogonale.

Pour rendre les scènes plus réalistes, les techniques d'illumination sont appliquées durant la phase de rendu en plaçant des sources de lumière dans la scène pour créer une atmosphère plus réaliste (Figure 1.16). La lumière dans les scènes possède des propriétés comme la couleur, la direction et l'intensité (force de la lumière). Il faut aussi prendre en compte les différents types de lumières comme par exemples les lumières ambiantes, diffuses et spéculaires.



(a) (b)

Figure 1.15 : Exemples d'animation 3D. (a) Animation de personnages. (b) Simulation des éffets phisyques et des phénomènes naturels.

Les modèles d'ombrage déterminent la manière dont la lumière est réfléchie ou réfractée par une surface en un point donnée de l'objet 3D et la manière avec laquelle les propriétés de ses matériaux varient sur ladite surface. Il faut déterminer au préalable les textures et les propriétés matériaux des objets 3D qui composent la scène et les techniques de placage de textures associées. Il faut aussi déterminer l'ambiance globale de la scène pour qu'elle soit prise en compte durant la phase de rendu. Afin d'augmenter le réalisme et simuler au maximum le monde réel, la tendance aujourd'hui est de pencher vers les rendus basés sur la physique, plus précisément, il s'agit d'une approche qui cherche à modéliser le comportement de la lumière dans le monde réel. Cette approche fournit une représentation plus précise de la façon dont la lumière interagit avec les propriétés des matériaux présents sur les différentes surfaces des objets 3D (Figure 1.17).

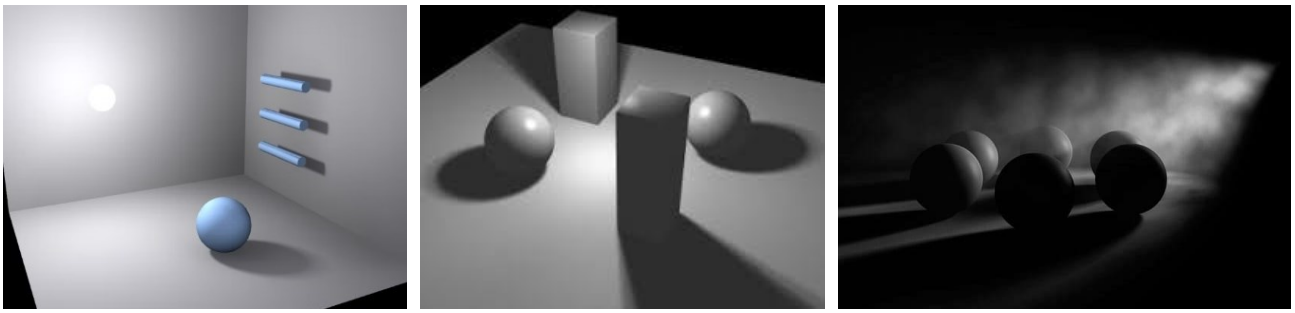


Figure 1.16 : Scènes contenant des objets 3D et des sources de lumière.



Figure 1.17 : Rendu basé sur la physique.

1.3.2 Le rendu

Le rendu (Figure 1.18) est l'étape au cours de laquelle la scène 3D est traduite en une image en appliquant des techniques comme le placage de textures, les techniques d'ombrage et d'illumination, les effets visuels, la déformation, la collision, les techniques de radiosité, les techniques à base de traçage de rayons, etc. Le processus de rendu des scènes 3D prend généralement beaucoup de temps et consomme beaucoup de ressources matérielles. Alors pour optimiser le traitement, deux critères sont considérés, la vitesse de rendu et la qualité visuelle des images. Rendre prioritaire l'un de ces facteurs au profit de l'autre dépend du type de rendu pouvant être en post-production ou en temps réel.

Le rendu post-production appelé également rendu différé, rendu précalculé ou rendu réaliste. Son principal objectif est la création des scènes 3D avec une très haute qualité. Dans ce type de rendu, les algorithmes mettent l'accent sur les différents aspects et détails de la scène, que ce soit au niveau de la modélisation des maillages 3D en augmentant leurs densités (plus de sommet et de polygones, surfaces NURBS, lissage…) ou au niveau du rendu lui-même dont les algorithmes associés (ombrage, illumination, traçage de rayons…) sont très complexes et nécessitent un temps considérable et qui s'exécutent durant plusieurs

passes. C'est pour cette raison que ce type de rendu n'est utilisé que dans les domaines ou l'interaction avec l'utilisateur n'est pas présente comme la création des séquences vidéo dans les domaines du cinéma et de la publicité.



(a) (b)

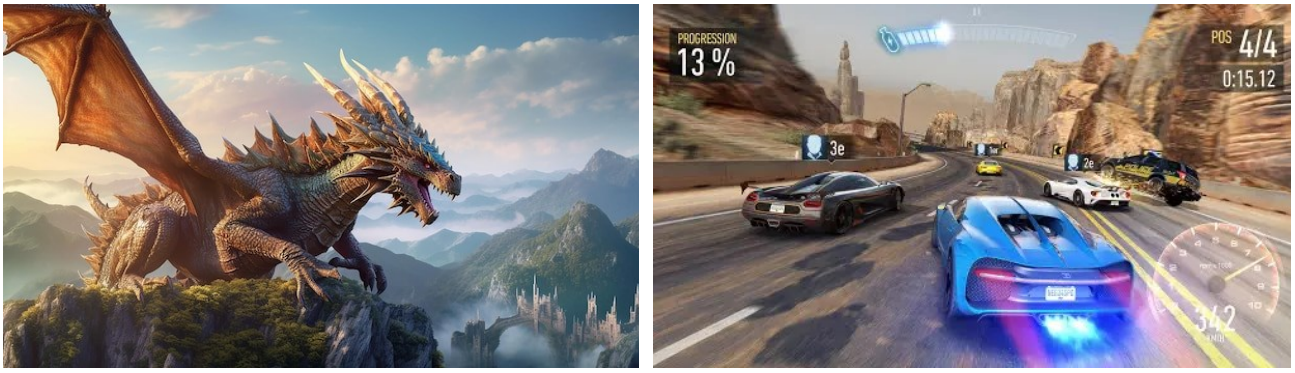
Figure 1.18 : Rendu d'une scène composée de différents objets 3D. (a) Rendu avec placage de texture seulement. (b) Rendu avec placage de texture et en appliquant les techniques d'illumination.

Le rendu temps réel quant à lui, mette l'accent plutôt sur la vitesse de traitement, car son objectif est la création des scènes 3D à une vitesse maximale dont l'animation est contrôlée par l'utilisateur. Dans ce type de rendu, les images sont calculées à très grande vitesse de sorte qu'il semble que les scènes, composées d'une multitude d'images, se produisent en temps réel lorsque les utilisateurs interagissent avec la scène. C'est pourquoi l'interactivité et la vitesse jouent un rôle important dans le processus de rendu temps réel. Par exemple, si vous souhaiteriez déplacer un personnage dans votre scène, vous devez vous assurer que le mouvement du personnage est mis à jour avant de dessiner l'image suivante, afin qu'il s'affiche avec une vitesse à laquelle l'œil humain peut percevoir un mouvement naturel. On trouve par exemple dans ce type de rendu, les jeux vidéo, les réalités virtuelles et les réalités augmentées.

Le rendu temps réel et le rendu précalculé s'utilisent parfois conjointement dans plusieurs situations car les domaines d'application ne se limitent pas exclusivement à l'un ou à l'autre. En effet, dans le domaine des jeux 3D, on trouve des séquences de vidéo introductifs ou explicatifs ou des séquences pour enchainer les stages des joueurs et leurs donner l'impression qu'ils font partie des personnages réels du jeu. Dans le domaine du cinéma, les infographistes ont besoin de prévisualiser les scènes et les objets qui les composent avant de calculer le rendu final, est cela s'effectue à travers le rendu en temps réel. De plus, plusieurs algorithmes et techniques s'utilisent dans les deux types de rendus. Même les deux types de rendu reposent sur des calculs sophistiqués, les logiciels et les bibliothèques 3D modernes peuvent offrir des fonctionnalités assez faciles à comprendre et à utiliser et permettent de réaliser des graphismes vraiment époustouflants. En outre, les architectures parallèles du matériel informatique et plus précisément, les cartes graphiques modernes ont aussi contribué à l'optimisation de la vitesse de rendu et à l'amélioration de la qualité d'images.

Au fil des ans, un certain nombre de solutions ont été proposées, on en trouve parmi d'autres la simplification du maillage. Cette technique est le processus qui consiste à prendre un modèle détaillé et à réduire le nombre de triangles tout en essayant de ne pas compromettre son apparence. Malgré cela, ce processus entraîne généralement la perte des

caractéristiques intéressantes des surfaces 3D, et l'objet perd son attrait visuel original. La raison qui a poussé les chercheurs et les ingénieurs à s'orienter vers une alternative au maillages polygonaux. Ceci a donné naissance aux techniques de modélisation et de rendu à base d'images (IBMR) qui deviennent de nos jours une alternative très importante aux techniques de rendu à base de polygones, puisqu'elles permettent potentiellement l'affichage des surfaces 3D en temps réel à un coût de performance optimale. La modélisation à base d'images utilise des textures pour stocker l'ensemble de données relatives à la géométrie qui seront récupérées pendant la phase de rendu en utilisant des algorithmes de traçage de rayons. Ces techniques permettent d'éviter de traiter un grand nombre de primitives graphiques (sommets et polygones), de plus, elles traitent seulement les pixels visibles contrairement au rendu à base de polygones qui utilise des algorithmes de test de visibilité consommant beaucoup de ressources.



(a) (b)

Figure 1.19 : Exemples de rendu de deux scènes 3D. (a) Rendu précalculé. (b) Rendu en temps réel.

1.4 PIPELINE GRAPHIQUE

En infographie, un pipeline graphique est un modèle conceptuel qui décrit les étapes qu'un système graphique doit effectuer pour restituer une scène 3D sur un écran 2D. Une fois qu'un modèle 3D est créé, par exemple dans un jeu vidéo ou toute autre animation informatique 3D, le pipeline graphique est le processus de transformation de ce modèle 3D en ce que l'ordinateur affiche.

Étant donné que les étapes nécessaires à cette opération dépendent des logiciels, du matériel utilisé et des caractéristiques d'affichage souhaitées, il n'existe pas de pipeline graphique universel adapté à tous les cas. Cependant, des interfaces de programmation d'applications graphiques (API) telles que Direct3D et OpenGL ont été créées pour unifier des étapes similaires et contrôler le pipeline graphique d'un accélérateur donné.

Afin de profiter pleinement du pipeline graphique moderne et étudier ces différentes étapes, il faut tout d'abord déterminer les composants principaux d'un système graphique. Il faut disposer d'une carte graphique de nouvelle génération. Ensuite, il faut étudier l'architecture du pipeline graphique adopté par les développeurs des bibliothèques graphiques (OpenGL, Vulkan, Direct3D, Metal, etc.) et les fabricants des cartes graphiques (NVIDIA, AMD, Intel) (voir *Annex A*).

Depuis leurs apparitions au début des années 2000, les cartes graphiques programmables ne cessent d'évoluer en offrant plus de flexibilité et plus de contrôle au niveau du développement des applications 3D, et cela à travers le pipeline graphique qui a été conçu en plusieurs étapes. Chacune de ces étapes assure un rôle spécifique dans la création de l'image finale et peuvent éventuellement être divisées en plusieurs sous-étapes. Les étapes conceptuelles du pipeline graphique sont présentées dans la Figure 1.20.

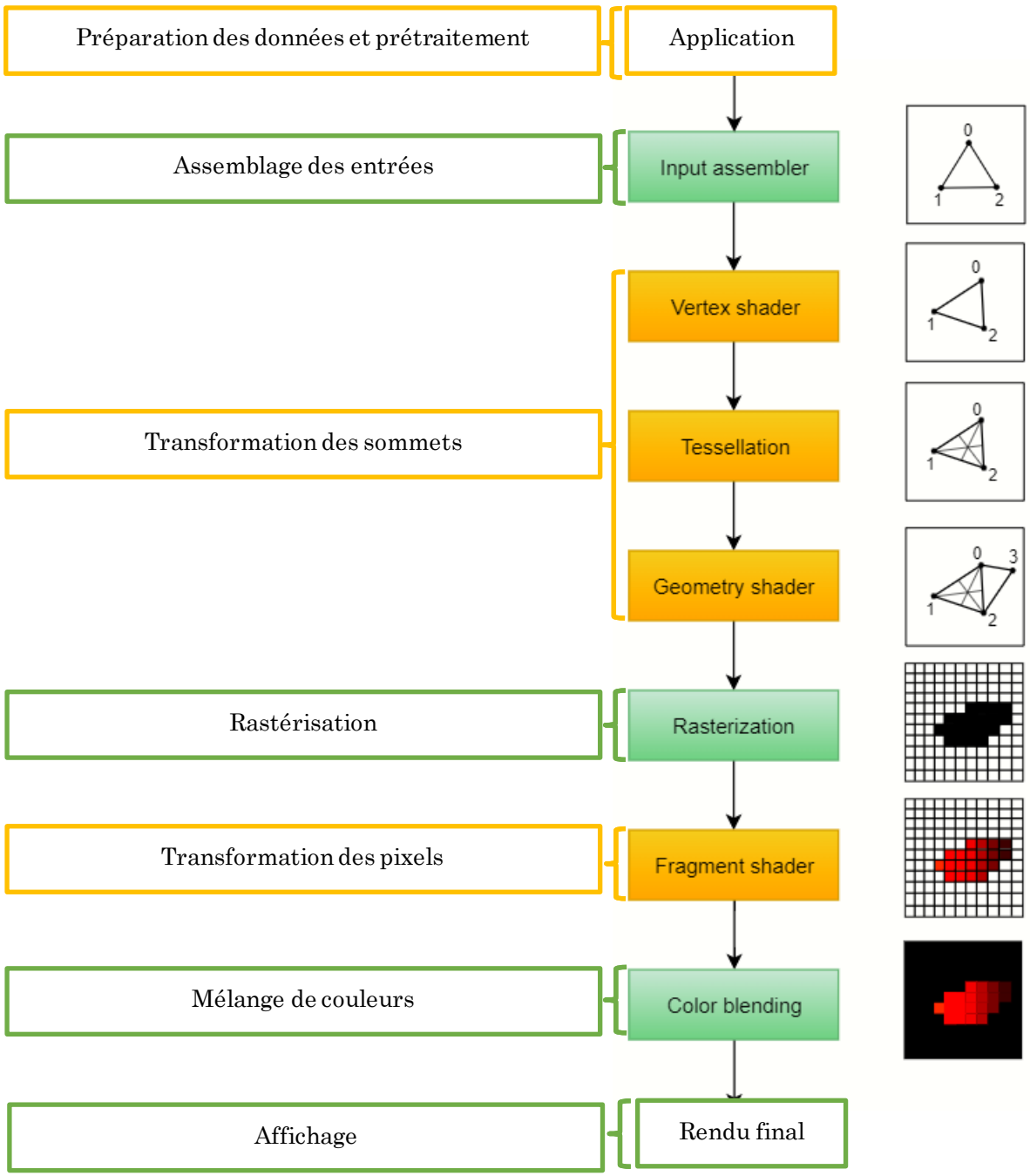


Figure 1.20 : Les grandes étapes du pipeline graphique. Les étapes programmables sont schématisées en jaune.

Ce pipeline conceptuel est implémenté et matérialisé physiquement par les fabricants des GPU modernes (NVIDIA, AMD, Intel) à l'aide des unités séparés au niveau des GPU. Ces unités peuvent être préprogrammées par les fabricants ou programmables par les développeurs d'applications 3D à travers les API en utilisant des programmes spécifiques appelés shaders. Ces unités sont structurées dans une chaine ou chaque unité couvre une

partie du pipeline en recevant ses données de l'unité précédente et en envoyant ses résultats à l'unité suivante (Figure 1.20).

Les Shaders sont des programmes utilisés pour gérer les algorithmes de rendu en temps réel dans les applications graphiques. Ils sont exécutés sur le GPU plutôt que sur le CPU pour maximiser les performances. Les Shaders sont utilisés pour effectuer des tâches telles que la modification de la couleur et de l'ombrage des pixels, le calcul de la silhouette, la déformation et la simulation des effets, des matériaux et de textures. Ils sont souvent écrits dans un langage de programmation spécifique, les plus connus sont :

- GLSL (OpenGL Shading Language) : utilisé avec OpenGL et Vulkan.
- HLSL (High-Level Shading Language) : utilisé avec Direct3D.
- Metal Shading Language : utilisé avec Metal.
- Cg (C for Graphics) : développé par NVIDIA et peut être utilisé avec plusieurs API 3D.

Dans cette section, nous allons décrire le pipeline graphique à travers les bibliothèques OpenGL et Vulkan et leurs langages de programmation à traitement parallèle GLSL.

1.4.1 Préparation des données et prétraitement

La préparation des données fait référence au processus de chargement des données 3D dans la mémoire vidéo (VRAM) pour le pipeline graphique telles que les modèles 3D, les textures, les animations et les scripts pour qu'ils soient préparés au rendu.

Un objet 3D est constitué de plusieurs sommets est d'un ensemble de primitives indiquant la manière avec laquelle ces sommets sont connectés entre eux. Les primitives sont les formes de base utilisées pour construire des objets graphiques complexes. Les plus couramment utilisées dans OpenGL (Figure 1.21) sont :

- Point : représente un sommet ou un pixel sur l'écran.
- Ligne : relie deux points entre eux.
- Triangle : utilisé pour relier trois points différents.
- Quadrilatère : ensemble de quatre points connectés utilisé souvent pour construire des surfaces planes.
- Tétraèdre : ensemble de quatre points connectés pour construire une forme en trois dimensions.

La grande majorité des modèles 3D sont conçus à base de triangles, c'est à dire, chaque trois sommets sont connectés entre eux à l'aide d'un triangle. On parle alors de maillage triangulaire.

Alors, dans cette étape, on détermine les positions des sommets et leurs normales, la primitive utilisée pour les connectés, les couleurs, les coordonnées de textures, etc. Dans le cas de l'utilisation d'une primitive de type polygone, les normales aux polygones peuvent être aussi déterminés. Ces données peuvent être préparés d'une manière procédurale ou être saisies à partir de différentes sources, telles que des fichiers stockés sur un disque dur, des

bases de données, des capteurs, etc. Les formats de fichiers couramment utilisés pour les données de modèles 3D incluent OBJ, FBX, COLLADA, etc. Les formats de fichiers couramment utilisés pour les textures incluent PNG, JPG, TGA, etc.

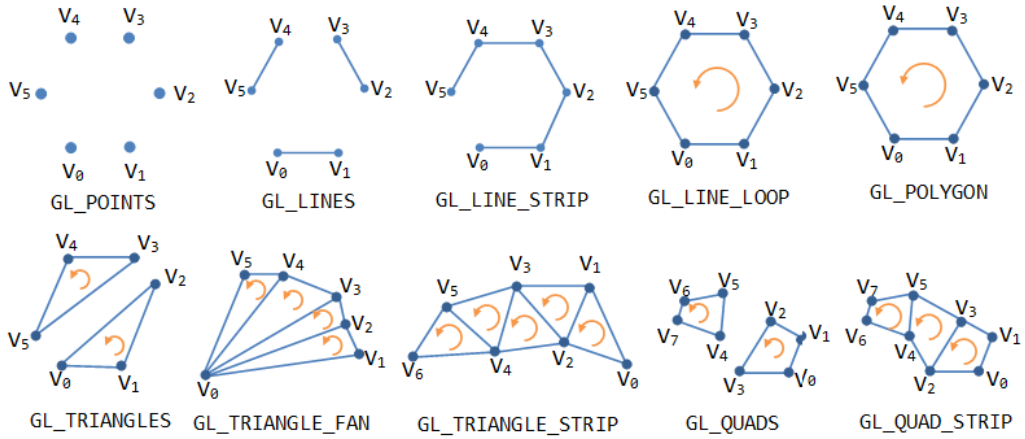


Figure 1.21 : Différentes primitives graphiques d'OpenGL pour connecter les sommets du maillage 3D.

Il est important que les données soient chargées efficacement pour éviter les ralentissements dans le pipeline graphique. Les données peuvent également être converties et optimisées pour s'adapter aux besoins spécifiques du pipeline graphique. Par exemple, les modèles 3D peuvent être triangulés pour être utilisés avec les algorithmes de rendu, des paramètres et des données supplémentaires (binormales, tangentes, surface quadratique, etc.) liés aux sommets peuvent être calculés, les textures peuvent être optimisées et compressées pour réduire la bande passante nécessaire pour les charger, etc. Il faut aussi déterminer le type de la caméra et ses caractéristiques (position, angle de vue, etc.).

1.4.2 Assembleur des entrées

Cette étape consiste à assembler les données 3D de l'application en un seul ensemble. Cela inclut la définition de l'ordre des triangles, des sommets, des normales, des textures, etc. Il peut également gérer des instanciations de modèle en utilisant les données supplémentaires pour générer plusieurs copies d'un même modèle en utilisant les mêmes données de sommets. Ces données sont stockées dans des tampons mémoire qui seront utilisés pour dessiner la scène.

L'assembleur des entrées peut également exploiter des informations de configuration pour décider comment les données doivent être utilisées pour dessiner la scène. Il peut également utiliser des informations de configuration pour décider comment ces données doivent être interprétées (par exemple, en utilisant des données de normales pour le calcul des ombres). L'assembleur des entrées est généralement implémenté par les bibliothèques et API 3D, il est donc une étape cruciale dans le pipeline graphique.

1.4.3 Transformation des sommets

La transformation des sommets fait référence au processus de modification de la position, de l'orientation, de l'échelle et de l'espace des objets dans une scène 3D. Elle fait référence aussi au processus de modification de la géométrie des objets ou aux maillages qui

les représentent. Les transformations sont généralement effectuées par des matrices qui sont multipliées aux coordonnées des objets. Il existe différents types de matrices pour différents types de transformations, tels que les matrices de translation pour déplacer les objets, les matrices de rotation pour les faire tourner, les matrices de mise à l'échelle, les matrices pour le changement d'espace, etc.

Il est important de noter que les transformations sont appliquées dans un ordre spécifique pour obtenir le résultat souhaité. Par exemple, si vous voulez déplacer un objet, puis le faire tourner, vous devez d'abord appliquer la transformation de translation, puis la transformation de rotation. Il est important de noter aussi que les transformations sont généralement effectuées sur des objets avant leur projection dans l'espace de la caméra, c'est pour cela qu'elles sont une étape clé dans le pipeline graphique. Les transformations sont généralement effectuées pour chaque objet individuellement, cela permet de conserver l'indépendance de chaque objet dans la scène.

Généralement, un modèle 3D est le plus souvent défini dans un espace qui lui est propre, appelé espace objet (repère de l'objet). Par exemple, les coordonnées des sommets d'une sphère sont définies par rapport à son centre. Cependant, la position, l'orientation et l'échelle de chaque objet sont définies par rapport à un espace commun appelé espace global (repère de la scène).

L'objectif principal de l'étape de transformation est de transformer les coordonnées tridimensionnelles des sommets (espace objet) en coordonnées normalisées (entre -1.0 et 1.0) en deux dimensions (espace écran) pouvant être affichées sur l'écran. Ces coordonnées sont ensuite passées à l'étape de rastérisation pour les transformer en coordonnées/pixels 2D sur l'écran. L'étape de transformation des coordonnées est associée à plusieurs espaces, chaque espace est défini par son propre système de coordonnées. Ils existent généralement cinq espaces par lesquels la transformation d'un objet 3D doit passer :

- **Espace local (ou espace objet) :** se réfère à un système de coordonnées qui est associé à un objet particulier dans un espace 3D (Figure 1.22a). Chaque objet peut avoir son propre système de coordonnées local, qui décrit sa position et sa forme relativement à ses propres axes X, Y et Z. L'utilisation d'un espace local permet de simplifier les calculs en travaillant dans un repère qui est attaché à l'objet lui-même, plutôt que dans un repère global où il peut y avoir beaucoup de transformations et de rotations à prendre en compte. Les transformations dans l'espace local peuvent être effectuées plus facilement et plus rapidement en utilisant des matrices de transformation simples, et peuvent être combinées pour décrire les transformations dans l'espace global.
- **Espace global** : est un espace unique associée à l'ensemble de la scène 3D. Il s'agit généralement d'un système de coordonnées fixe, qui décrit la position et la forme de tous les objets dans l'espace par rapport à un repère fixe (Figure 1.22b). L'espace global est souvent utilisé pour décrire les relations entre les objets dans un environnement plus large, telles que les positions des objets par rapport à d'autres objets ou à une référence absolue telle que l'origine du système de coordonnées. Les transformations dans l'espace global peuvent être plus complexes et nécessitent des matrices de

transformation pour décrire les rotations et les translations, ces matrices sont appelées des matrices de Model.

- **Espace de vue (ou espace œil)** : utilisé pour décrire la position et la forme des objets par rapport à une caméra virtuelle (Figure 1.22c). Il permet de simuler la vue d'un observateur depuis un point de vue spécifique. Cet espace est défini par une matrice appelée matrice de Vue qui prend en compte la position et l'orientation de la caméra, ainsi que les paramètres de vue tels que le champ de vision et la distance de vue.

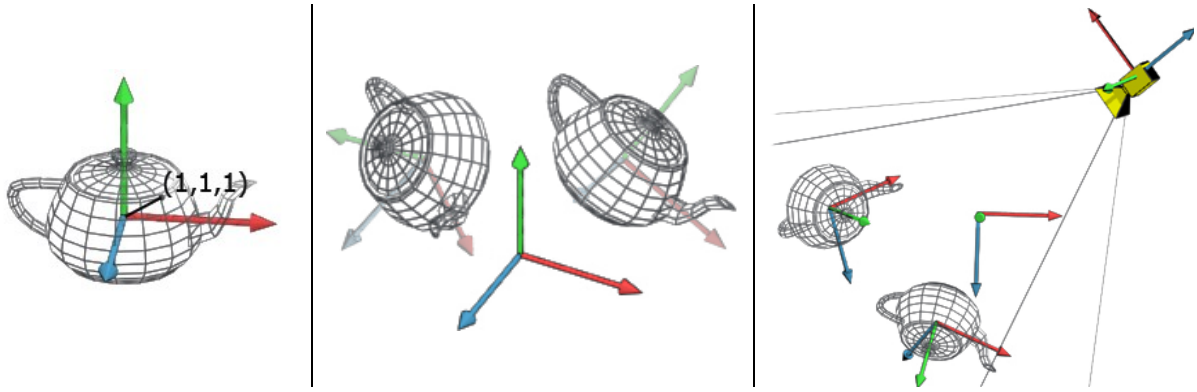


Figure 1.22 : Représentation d'une théière 3D dans différents espaces. (a) Espace local. (b) Espace globale. (c) Espace de vue.

- **Espace écran** : utilisé pour décrire la position des objets sur un écran 2D, tels qu'un moniteur d'ordinateur ou un écran de télévision. Les coordonnées dans cet espace sont définies par rapport à la taille de l'écran, généralement en utilisant des pixels comme unité de mesure. L'espace écran se concentre sur les objets qui doivent être affichés à l'écran, et les transformations dans cet espace sont généralement utilisées pour effectuer des opérations telles que le redimensionnement et le déplacement de l'affichage. Les objets dans l'espace de vue sont projetés sur l'écran en utilisant une matrice de Projection pour produire des coordonnées dans l'espace écran.

- **Espace de clip** : est un système de coordonnées pour déterminer les objets qui doivent être affichés à l'écran et ceux qui doivent être ignorés. Il se concentre sur les objets qui se trouvent à l'intérieur d'un certain volume, appelé volume de vue, et ignore ceux qui sont en dehors (Figure 1.23). Ce volume de vue est défini par la matrice de Vue et la matrice de Projection. Les objets dans l'espace globale sont projetés sur l'écran en utilisant ces matrices, et seuls ceux qui se trouvent à l'intérieur du volume de vue sont affichés.

La Figure 1.24 décrit le processus de transformation d'un objet 3D de l'espace local vers l'espace écran.

L'étape de transformation des sommets est une étape programmable qui peut être divisée en trois sous étapes implémentées dans trois Shaders différents qui sont *Vertex Shader*, *Tessellation Shader* et *Geometry Shader*.

- **Vertex Shader** : exécuté pour chaque sommet et applique généralement des transformations pour faire passer les positions des sommets de l'espace modèle à l'espace écran. D'autres opérations liées aux sommets peuvent être effectuées comme le

calcul des vecteurs normaux et des vecteurs tangents, détermination de la direction de la lumière et la direction de vu, calcul de l'espace tangent, etc.

- **Tessellation Shader** : utilisé pour ajouter de la complexité à la géométrie d'un objet 3D. Il permet de diviser la surface en plusieurs petits triangles (augmenter la densité du maillage) pour produire une représentation plus détaillée et plus lisse de la forme. Il peut être utilisé pour créer des formes plus complexes, telles que des courbes, des ondulations et des surfaces en relief, sans avoir à stocker de plus grandes quantités de données géométrique.
- **Geometry Shader :** permet de traiter les primitives graphiques (points, lignes, triangles) avant qu'elles ne soient rendues à l'écran. Une fois ces primitives reçues, il peut les modifier en les déplaçant, les redimensionnant, les faisant tourner ou les filtrant en ne sélectionnant que celles qui satisfont certaines conditions. Il peut également générer de nouvelles primitives géométriques, par exemple, il peut transformer un triangle en plusieurs triangles pour produire un effet de fumée ou de particules.

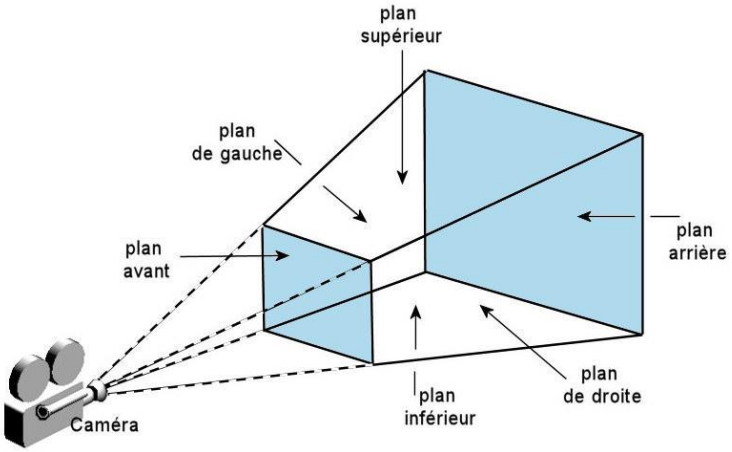


Figure 1.23 : Le volume délimité par les plans de la figure est appelé volume de vue.

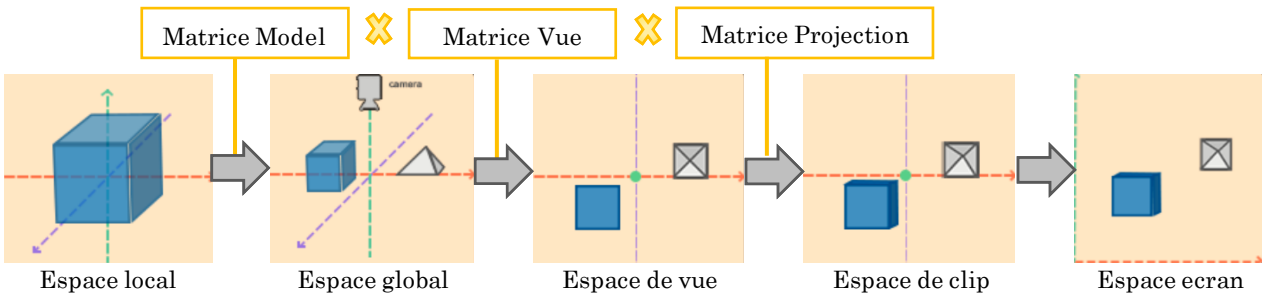


Figure 1.24 : Processus de transformation d'un objets 3D de son espace local à l'espace écran.

1.4.4 Rastérisation

La rastérisation est un processus complexe qui utilise des algorithmes mathématiques avancés pour convertir les primitives graphiques en pixels pour l'affichage à l'écran. Cela permet de déterminer quelles parties de l'image seront visibles à l'écran. Les calculs de profondeur et les tests de visibilité sont effectués dans cette étape afin d'éviter les effets de superposition indésirables. Pendant cette étape, le GPU interpole les données associées aux

sommets, comme la profondeur, la couleur, la normale ou encore les coordonnées de texture pour former une entité appelée *fragment*.

La rastérisation est divisée en plusieurs sous-étapes, durant lesquelles plusieurs tests sont effectués pour déterminer quelles parties de la scène doivent être affichées à l'écran. On trouve entre autres :

- Test de visibilité système : permet de traiter que les pixels appartenant à des fenêtres systèmes visible à l'utilisateur, c'est-à-dire, si une autre fenêtre du système est devant un pixel donné, il sera ignoré.
- Test de ciseaux : permet d'éliminer les pixels se trouvant hors d'un rectangle défini par l'application.
- Test alpha : permet de traiter que les pixels ayant une certaine valeur de transparence.
- Test de pochoir (Stencil) : permet d'éliminer certains fragments basés sur des fragments d'autres objets dessinés dans la scène selon des critères définis par l'utilisateur.
- Test de profondeur : permet d'éliminer les fragments selon leurs valeurs de profondeur. La profondeur détermine la distance entre le fragment et la caméra, l'ensemble de ces profondeurs sont stockées dans un tampon mémoire appelé Z-buffer.

1.4.5 Transformation des pixels

La transformation des pixels consiste à appliquer des transformations géométriques et/ou des effets visuels aux pixels d'une image pour modifier leur position, leur échelle, leur rotation, leur distorsion, leurs couleurs ou d'autres propriétés visuelles. Les étapes les plus importantes de la transformation des pixels sont détaillés comme suit :

- Récupération des coordonnées de l'écran : une fois que les objets 3D ont été transformés en coordonnées d'écran lors de l'étape de rastérisation, les pixels correspondants de l'image sont identifiés par leurs coordonnées d'écran (x, y).
- Échantillonnage des valeurs de texture : si l'image finale doit être texturée, les valeurs de texture correspondantes aux coordonnées d'écran peuvent être échantillonnées à partir d'une texture associée à l'objet 3D. Cela permet d'obtenir les couleurs, les motifs ou les détails de texture à appliquer aux pixels.
- Calcul des transformations géométriques : les pixels peuvent subir diverses transformations géométriques, telles que la translation, la rotation, l'échelle, la distorsion ou d'autres transformations spécifiques. Ces transformations sont généralement appliquées en utilisant des matrices de transformation qui transforment les coordonnées d'écran des pixels pour obtenir leurs nouvelles positions et attributs.
- Application des effets visuels : les effets visuels, tels que les filtres d'image, les effets de déformation, de particules, de lumière, de fumée, etc., sont appliqués aux pixels en fonction de leurs nouvelles positions et attributs. Cela peut impliquer des calculs complexes pour simuler des phénomènes visuels réalistes ou pour créer des styles artistiques spécifiques.

- Calcul des valeurs de couleur finales : une fois que les transformations géométriques et les effets visuels ont été appliqués aux pixels, leurs nouvelles valeurs de couleur sont calculées en combinant les valeurs de texture échantillonnées, les couleurs d'objet transformées et les effets visuels appliqués. Cela peut inclure des opérations de mélange de couleurs, d'opacité, de lumière et d'autres calculs pour obtenir les valeurs de couleur finales des pixels.
- Stockage des valeurs de couleur : les valeurs de couleur finales des pixels sont stockées dans un tampon d'image qui représente l'image finale à afficher.

Il convient de noter que la transformation des pixels est une étape programmable à travers le *Fragment Shaders* et peut être réalisée en utilisant des techniques et des algorithmes différents en fonction des API 3D, de la plateforme matérielle, du rendu en temps réel ou précalculé, et des exigences spécifiques de l'application graphique.

1.4.6 Mélange de couleurs et affichage

Le mélange de couleurs est une étape supplémentaire qui consiste à combiner les couleurs de différentes surfaces qui se superposent pour simuler des effets spéciaux tels que la transparence, la réflexion et la réfraction. Il utilise des algorithmes de mélange de couleurs qui exploitent des paramètres tels que les coefficients de transparence, les coefficients de réflexion afin déterminer comment les couleurs doivent être mélangées.

L'affichage est l'étape finale du pipeline graphique où les images générées sont affichées à l'écran. Il est important de noter que l'affichage ne se limite pas à l'affichage sur un écran d'ordinateur. Il peut également inclure l'affichage sur des écrans de télévision, des projecteurs, des écrans tactiles, des dispositifs de réalité virtuelle ou augmentée, etc.

1.5 DOMAINES D'APPLICATION

L'infographie 3D est un outil puissant qui est largement utilisé dans de nombreux domaines pour produire des images, des animations et des modèles en trois dimensions. Cet outil permet de visualiser des images et des modèles de manière claire et précise, ce qui peut aider les professionnels à prendre des décisions informées et à mieux comprendre les concepts complexes. Voici en détail certains des domaines d'application les plus courants de l'infographie 3D :

Divertissements : dans l'industrie des jeux vidéo, l'infographie 3D est utilisée pour créer des mondes virtuels, des personnages, des objets et des scènes de jeu (Figure 1.25). Les développeurs peuvent utiliser des logiciels pour modéliser les différents éléments du jeu, puis les animer pour donner des vies aux personnages et aux objets. L'infographie 3D est largement utilisée aussi dans l'industrie du cinéma pour produire des films d'animation et des séquences animées pour la télévision.

Architecture et conception : dans l'industrie de la construction et de la conception de produits, l'infographie 3D est utilisée pour créer des maquettes numériques d'immeubles, de

bâtiments et de produits industriels (Figure 1.26). Les architectes et les ingénieurs peuvent utiliser des logiciels d'infographie 3D pour visualiser les projets avant de les construire, ce qui peut aider à identifier les problèmes potentiels et à faire des ajustements avant la construction réelle.

Médecine : dans l'industrie médicale, l'infographie 3D est utilisée pour visualiser des modèles anatomiques en 3D pour la formation médicale et la planification chirurgicale (Figure 1.27). Les professionnels de la santé peuvent expliquer les conditions médicales complexes, les traitements disponibles, les effets secondaires des médicaments à travers des techniques 3D. Ils peuvent aussi inclure des illustrations pour représenter les organes et les systèmes du corps, des diagrammes pour expliquer les processus médicaux, etc. L'infographie 3D peut être utilisée aussi pour représenter les images médicales en 3 dimensions pour aider les médecins à diagnostiquer et à traiter les maladies.

Sciences et Éducation : dans l'industrie de l'éducation, l'infographie 3D est utilisée pour créer des simulations et des visualisations pour aider à enseigner des concepts complexes (Figure 1.28). Les enseignants peuvent utiliser des techniques 3D pour concevoir des simulations interactives pour les étudiants, ce qui peut aider à renforcer leur compréhension des sujets difficiles en science, en histoire, en géographie, etc. Les professeurs peuvent également utiliser l'infographie 3D pour créer des présentations interactives pour leurs cours. Ils peuvent aussi représenter les données géologiques en 3 dimensions pour montrer les reliefs, les rivières, les lacs et les autres caractéristiques du terrain.

Services spatiaux : dans l'industrie des services spatiaux, l'infographie 3D est utilisée pour concevoir et simuler des missions spatiales (Figure 1.29). Les ingénieurs spatiaux peuvent utiliser des logiciels d'infographie 3D pour concevoir des satellites, des fusées et d'autres équipements, puis les tester virtuellement pour vérifier leur fonctionnement.



Personnages Marvel



Street Fighter



Assassin Creeds



Star Wars

Figure 1.25 : Jeux et films d'animations populaires crées avec les techniques modernes de l'infographie 3D.

Technologies de l'information : les entreprises de technologie utilisent souvent des graphiques pour présenter les données complexes de manière claire et concise (Figure 1.30). Les graphiques peuvent inclure des diagrammes pour représenter les réseaux informatiques, les tendances technologiques, des illustrations pour expliquer les fonctionnalités des produits technologiques, etc.

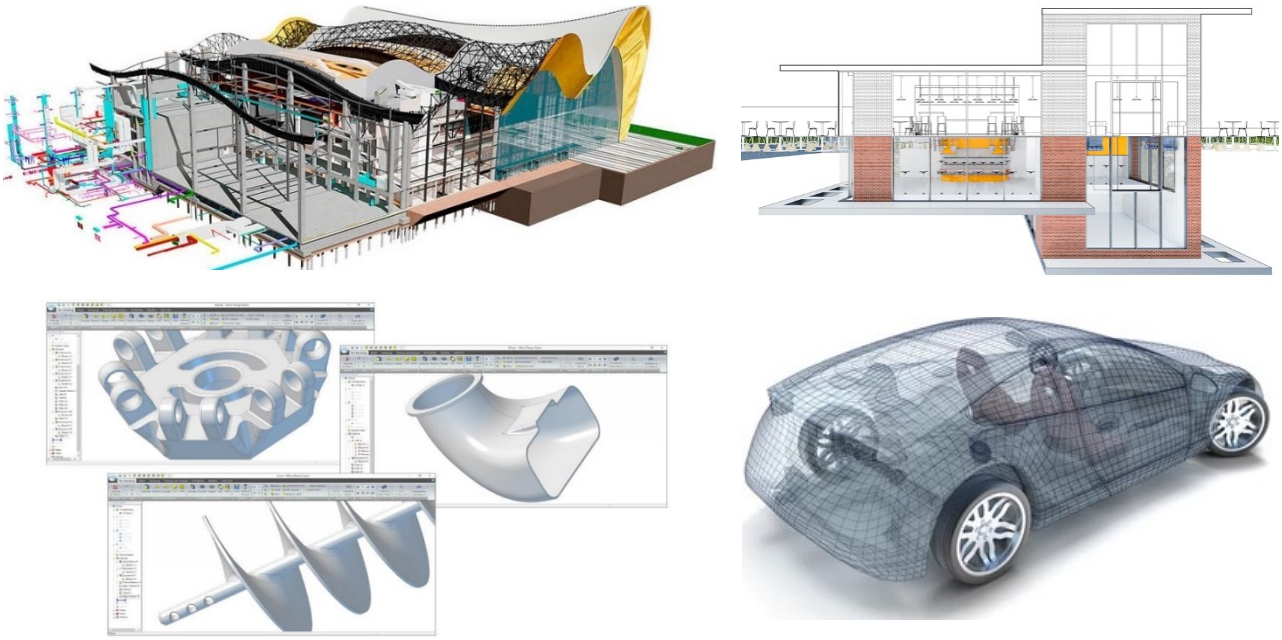


Figure 1.26 : Exemples d'architectures et de conception crées par ordinateur.

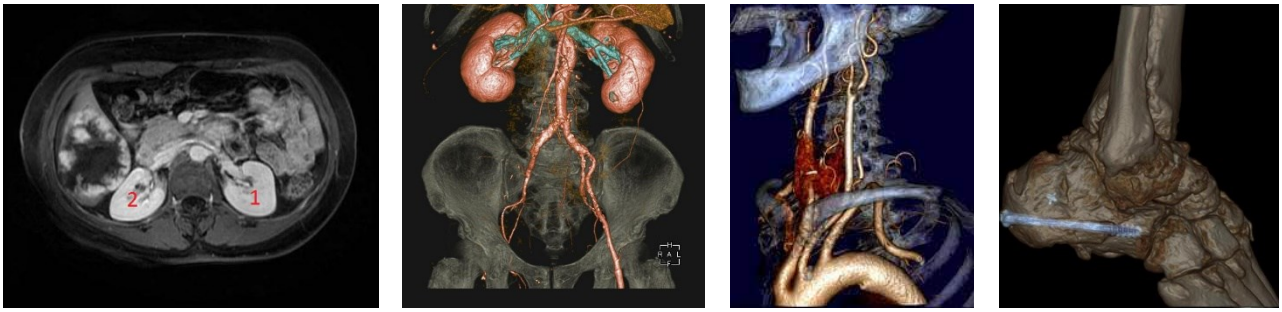


Figure 1.27 : Représentation des organes du corps humain en utilisant l'infographie 2D/3D.

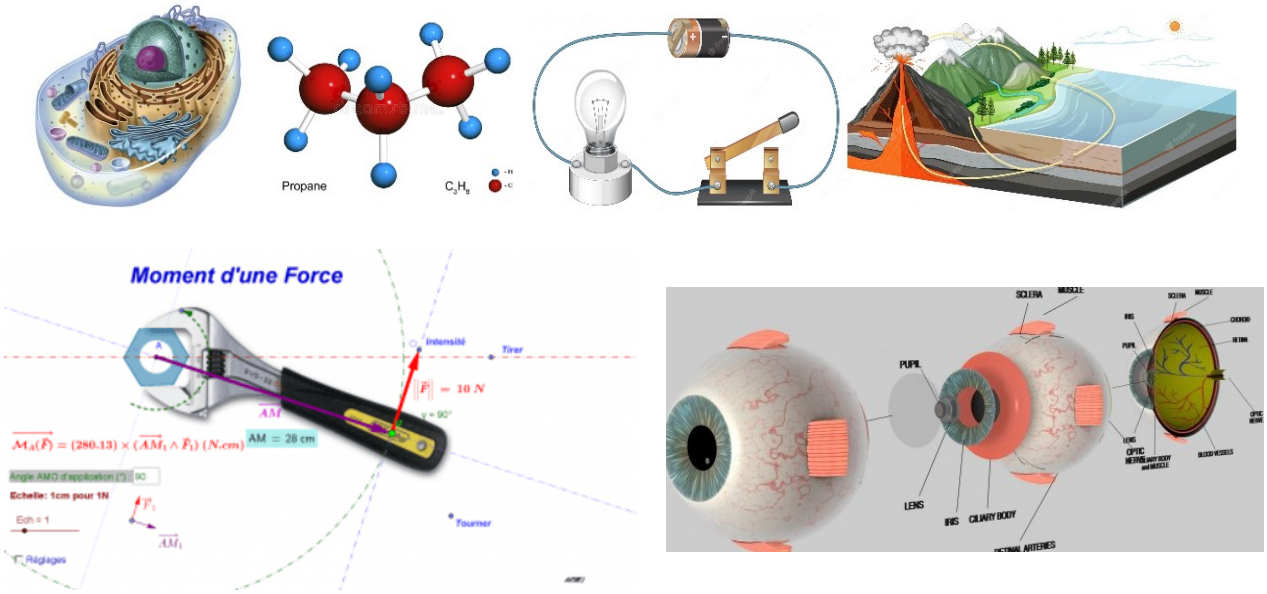


Figure 1.28 : Utilisation de l'infographie dans les domaines de l'éducation et la science.

Médias : les journaux, les magazines, les chaines TV et les sites web utilisent souvent des graphiques, des diagrammes et des illustrations pour représenter des informations telles que les statistiques économiques, les tendances politiques, les résultats sportifs, etc. (Figure 1.31). L'objectif est d'attirer l'attention des visiteurs et des spectateurs d'une manière attractive. A travers ces media, la publicité et le marketing exploitent les techniques 3D pour des présentations de produits et des séquences de vidéos publicitaires.

Il existe bien d'autres domaines de l'application de l'infographie tels que l'environnement, l'urbanisme, l'économie, la simulation de la vie quotidienne, le désigne de l'intérieur, la communication, réalité virtuelle et de réalité augmentée, etc. L'infographie 3D continue de se développer et de trouver de nouvelles applications dans de nombreux domaines, et son utilisation continue de s'étendre pour produire des images plus immersives et plus réalistes.

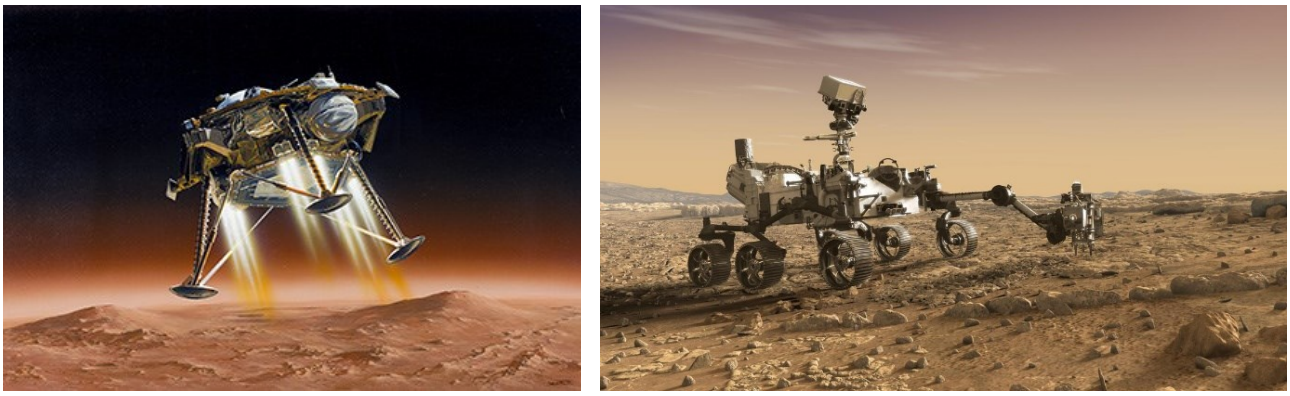


Figure 1.29 : Simulation 3D d'une mission sur Mars.

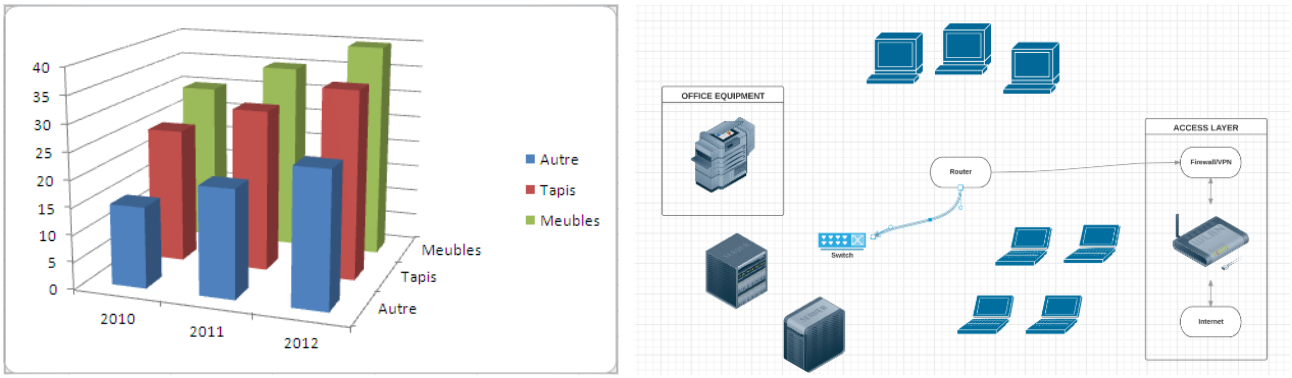


Figure 1.30 : L'utilisation de l'infographie dans le domaine de la technologie de l'information.



Figure 1.31 : Utilisation de l'infographie dans les médias.

1.6 CONCLUSION

L'infographie 3D est un outil puissant pour la représentation des informations qui continue d'évoluer pour répondre aux besoins des utilisateurs qui cherchent des moyens plus efficaces pour visualiser les données et de communiquer des informations. L'histoire de l'infographie a montré aussi une évolution continue de la manière dont les informations sont présentées et diffusées, en passant des illustrations manuelles aux graphiques complexes créés par ordinateur.

Aujourd'hui, l'infographie est considérée comme un outil indispensable pour la communication d'informations complexes, et les professionnels de l'infographie sont en demande dans de nombreux domaines. Avec l'avènement de l'Internet et du Web, la popularité croissante des médias sociaux (comme Facebook, Twitter, Instagram, YouTube, etc.), les outils en ligne et les logiciels de création d'infographies ont considérablement simplifié la création des graphiques pour les non-professionnels et ont permis aussi à un large public d'accéder à des informations graphiques de manière plus rapide et plus facile.

L'avenir de l'infographie est incertain, mais il est probable que la technologie et les outils continueront d'évoluer pour permettre la création des graphiques plus attractives. L'intelligence artificielle et la réalité augmentée pourraient également jouer un rôle important dans l'avenir de l'infographie, en permettant aux utilisateurs de visualiser les données de manière encore plus réalistes et interactive.

L'objectif de ce chapitre était juste l'exposition des principes de base et les étapes fondamentales qui contribuent à la création des scènes 3D en exploitant les architectures parallèles des carte graphiques modernes. Cela permettra par la suite de mieux comprendre les concepts du placage de déplacement tridimensionnel par pixel et les contributions apportées qui rentre eux-mêmes dans le cadre des techniques de placage de textures. Généralement, ces techniques sont toujours combinées avec des techniques d'illumination, notamment avec des modèles d'ombrage afin de donner l'aspect du réalisme aux objets 3D. Les détails concernant ces derniers principes sera l'objectif du chapitre suivant.

Chapitre 2

MODELES D'OMBRAGE ET PLACAGE DE TEXTURES

2.1 INTRODUCTION

L'apparence des surfaces géométriquement complexes est un défi majeur pour le rendu en temps réel, notamment dans le cas où les microreliefs couvrent la surface d'un objet 3D. Ces derniers nécessitent d'énormes ensembles de primitives et de polygones pour les modéliser, et cela entrave énormément le processus de rendu de ces surfaces et spécialement les applications qui nécessitent un rendu à des fréquences d'images interactives.

L'apparence de ces microreliefs est nécessaire puisqu'ils produisent des effets visuels importants tels que l'auto-ombrage, l'occlusion et la silhouette. En combinant les modèles d'ombrage et les techniques de placage de textures, on peut créer des objets 3D très réalistes tout en optimisant la complexité des algorithmes de rendu. Afin de mettre en évidence cette apparence, les objets 3D sont rendus en prenant en compte l'éclairage de la scène qui est un élément important pour créer des images convaincantes et approcher les scènes 3D du monde réel. L'éclairage dans les scènes 3D est assuré par des sources de lumière qui peuvent être de différents types et de différentes couleurs. Ces sources de lumière sont prises en compte par les modèles d'ombrage dans la phase de rendu au niveau du pipeline graphique.

Les modèles d'ombrage sont des algorithmes qui simulent l'interaction de la lumière avec les surfaces des objets pour produire des reflets réalistes. Le placage de textures, quant à lui, est une technique qui consiste à appliquer une image sur la surface d'un objet pour lui donner un aspect réaliste. Les textures peuvent être utilisées pour simuler des matériaux tels que le bois, le métal, le tissu, etc.

L'objectif de ce chapitre est de décrire les concepts de base qui contribuent à l'apparence des objets 3D, notamment, les modèles d'ombrage et les techniques de placage de textures. Il abord les sections suivantes :

- Une description globale de la structure d'un objets 3D avec les éléments de base qui le compose.
- Les principes et les composants de l'illumination dans les scènes 3D.
- Les différents modèles d'ombrage.
- Les différentes techniques de placage de texture qui contribuent à l'amélioration de l'apparences des objets 3D.

2.2 STRUCTURES DES OBJETS 3D

Les objets 3D peuvent être décrits par plusieurs types de structures géométriques (Figure 2.1), qui dépendent du contexte d'utilisation, comme par exemples :

– Maillage triangulaire : est une structure de données qui représente un objet 3D sous la forme d'un ensemble de triangles connectés. Chaque triangle est défini par trois points dans l'espace 3D, appelés sommets.

- Voxel : est un élément de volume, similaire à un pixel dans une image 2D. Ils sont utilisés pour représenter des objets en trois dimensions dans des environnements de rendu volumétrique.
- Nuage de points : est une collection de points dans l'espace 3D, qui peuvent représenter les contours ou les surfaces d'un objet. Les nuages de points sont souvent générés par des scanners 3D et sont utilisés pour créer des modèles 3D précis.
- Courbes et surfaces : les courbes et les surfaces (Bézier, Spline) sont des structures mathématiques qui permettent de décrire des objets en trois dimensions à l'aide de formes géométriques simples.
- Etc.

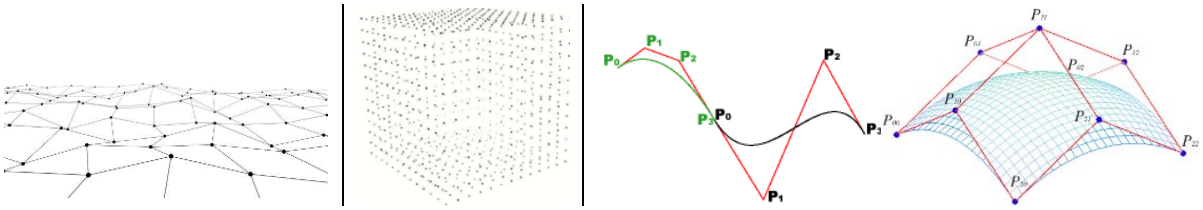


Figure 2.1 : Exemple de structures géométriques pouvons décrire un objet 3D.

L'apparence d'un objet 3D est l'un des aspects les plus importants de sa représentation. Elle décrit les propriétés visuelles de l'objet, telles que la couleur, la texture, la transparence, la réflexion et la réfraction. Généralement, l'apparence des modèles 3D est constituée d'une structure à trois niveaux, ces derniers peuvent être définis comme suit :

- Macrostructure : décrit la structure géométrique globale de l'objet 3D ainsi que ses grandes caractéristiques.
- Microstructure : est le niveau qui permet de déterminer la couleur et les propriétés des matériaux de l'objet 3D (bois, métal, plastique, pierre, eau, feux, verre, etc.)
- Mesostructure : est le niveau de la structure qui se situe entre la macrostructure et la microstructure. Il décrit les caractéristiques de la structure à une échelle intermédiaire, notamment, les microreliefs qui couvrent la surface de l'objet. La mesostructure peut affecter les propriétés physiques et mécaniques du matériau en créant des gradients de propriétés à travers l'objet.

Généralement, les modèles d'ombrage interviennent au niveau des microstructures et les techniques de placage de textures interviennent au niveau des mesostructure.

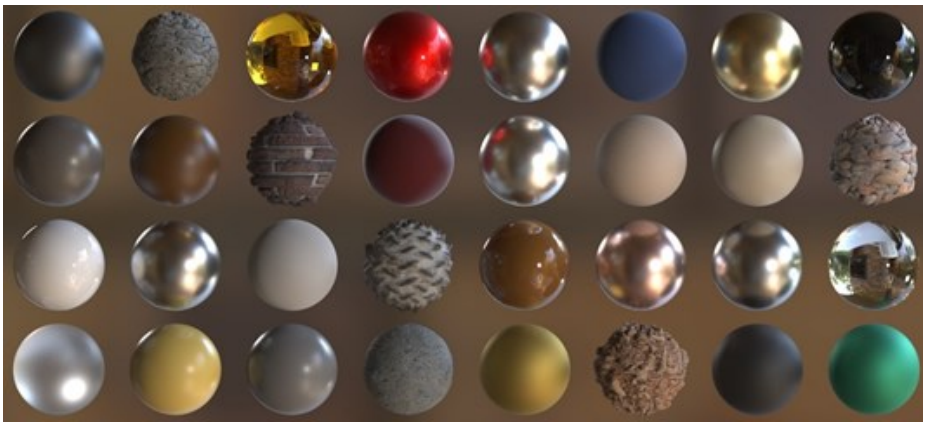


Figure 2.2 : Des objets 3D rendus avec différentes techniques de placage de texture et différentes techniques d'ombrage.

2.3 L'ECLAIRAGE DANS LES SCENES 3D

L'éclairage est un élément clé en infographie 3D car il permet de donner du réalisme à la scène créée. Lorsqu'on parle d'éclairage dans une scène, on fait référence à la simulation de la façon dont la lumière interagit avec les objets présents dans la scène, en prenant en compte les propriétés physiques des matériaux et de la lumière elle-même. L'éclairage peut être utilisé pour créer des ambiances et des atmosphères différentes, en jouant sur les couleurs, les lumières et les reflets. Il peut également être utilisé pour mettre en valeur certains éléments de la scène, en les éclairant plus intensément que le reste de la scène.

L'éclairage dans les scènes repose sur trois éléments principaux qui sont les couleurs, les matériaux et les sources de lumières. La couleur peut avoir un impact important sur l'ambiance de la scène. Par exemple, une lumière bleue peut donner une ambiance froide tandis qu'une lumière jaune peut donner une ambiance chaleureuse. Les sources de lumière identifient les sources des rayons lumineux dans la scène et leur type (soleil, lampe, bougie, etc.), chaque type de source lumineuse aura des propriétés différentes qui influenceront l'apparence de la scène. Les matériaux déterminent les propriétés physiques des objets 3D qui affectent la façon dont la lumière est réfléchie, réfractée ou absorbée.

Il est important de bien comprendre comment l'éclairage peut affecter la scène 3D, car cela peut faire la différence entre une scène qui paraît naturelle et une scène qui paraît artificielle.

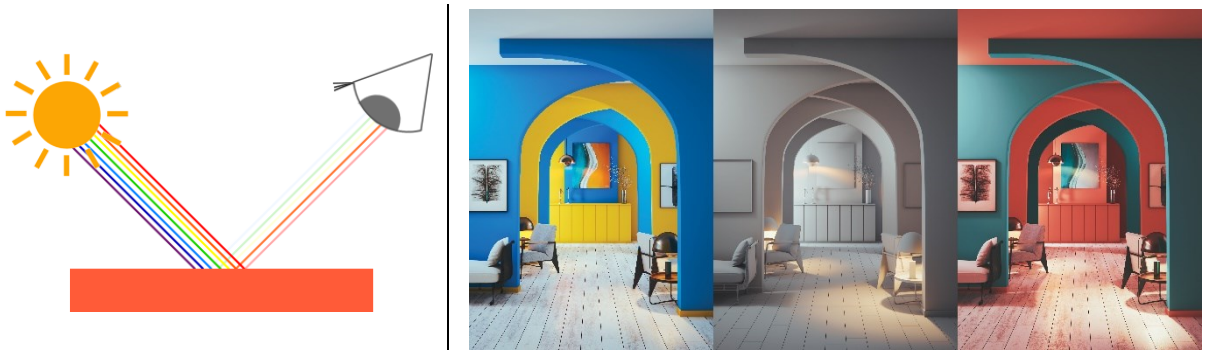
2.3.1 Les couleurs

L'éclairage repose sur le calcul des couleurs des pixels d'une image en fonction de la position de la source de lumière, de l'objet et de l'observateur. Plus précisément, le processus d'éclairage implique le calcul de la quantité de lumière qui atteint chaque pixel de l'objet en fonction de son orientation par rapport à la source de lumière. Cette quantité de lumière est ensuite utilisée pour calculer la couleur finale de chaque pixel de l'objet en fonction des propriétés de sa surface (comme sa réflectivité et sa texture).

Les couleurs sont des éléments importants car elles contribuent grandement à l'apparence et à l'ambiance générale de la scène. Les couleurs peuvent être utilisées pour indiquer la texture, la réflexion, la brillance, l'ombre et la luminosité d'un objet, etc. Dans le monde réel, les couleurs peuvent prendre n'importe quelle valeur de couleur connue, chaque objet ayant sa ou ses propres couleurs.

Dans le monde numérique, nous devons mapper les couleurs réelles (infinies) sur des valeurs numériques (limitées) et, par conséquent, toutes les couleurs du monde réel ne peuvent pas être représentées numériquement. Les couleurs sont représentées numériquement à l'aide des composantes rouge, vert et bleu communément abrégé en RVB. En utilisant différentes combinaisons de ces 3 valeurs seulement, dans une plage de [0,1], nous pouvons représenter presque toutes les couleurs existantes.

La couleur d'un objet que nous voyons dans la vraie vie n'est pas la couleur qu'il a réellement, mais la couleur réfléchie par celui-ci. Les couleurs qui ne sont pas absorbées (rejetées) par l'objet sont les couleurs que nous en percevons (Figure 2.3b). Par exemple, la lumière du soleil est perçue comme une lumière blanche qui est la somme combinée de nombreuses couleurs différentes (Figure 2.3a). Si nous faisions briller cette lumière blanche sur un objet bleu, il absorberait toutes les sous-couleurs de la couleur blanche à l'exception de la couleur bleue. Étant donné que l'objet n'absorbe pas la partie de couleur bleue, celle-ci est réfléchie et pénètre notre œil, ce qui donne l'impression que l'objet a une couleur bleue.

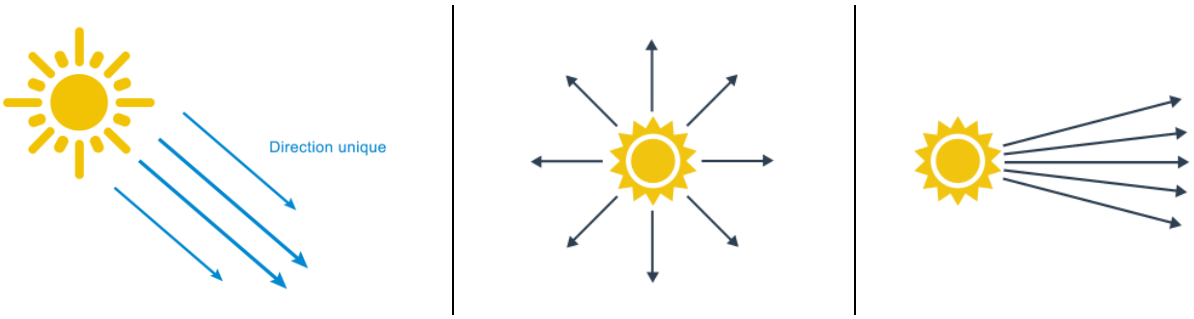


(a) (b)

Figure 2.3 : L'effets de la couleur sur l'œil humain (a) et sur l'apparences des scènes 3D (b).

2.3.2 Les sources de lumière

La lumière est un élément fondamental dans les scènes 3D car elle permet de créer des ombres, des reflets, des effets de brillance et de donner de la profondeur à la scène. En infographie 3D, ils existent plusieurs types de lumières (Figure 2.4) dont les principaux sont : les lumières directionnelles (soleil), les lumières ponctuelles (effets de lampe), et les lumières volumétriques (effet projecteur).



(a) (b) (c)

Figure 2.4 : Les trois types de lumière les plus utilisées en infographie 3D.(a) Lumière directionnelle. (b) Lumière ponctuelle. (c) Lumière volumétrique.

2.3.3 Les lumières directionnelles

Les lumières directionnelles sont un type de lumière communément utilisé pour simuler des conditions de lumière naturelles, comme la lumière du soleil ou la lumière de la lune en soirée (Figure 2.5a). Elles sont souvent utilisées pour créer une ambiance générale dans la scène, ainsi que pour donner de la profondeur et du contraste en créant des ombres. Comme leur nom l'indique, les lumières directionnelles ont une direction et une intensité définies, et donnent une lumière uniforme dans cette direction.

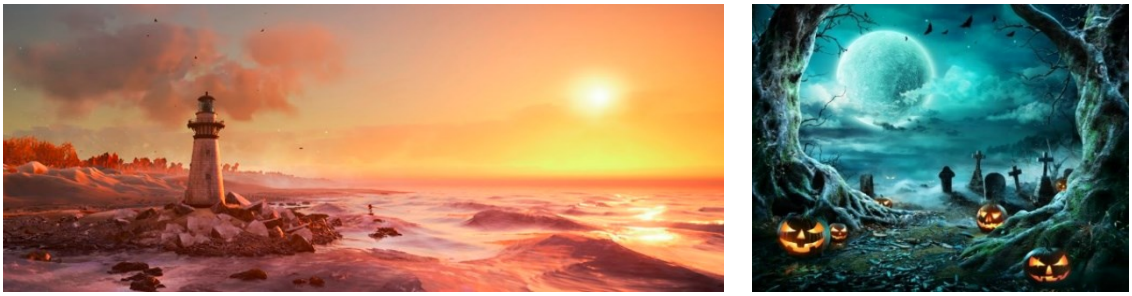


Figure 2.5 : Scènes 3D rendues avec une lumière directionnelle.

2.3.3.1 Les lumières ponctuelles

Les lumières ponctuelles sont un autre type de lumière couramment utilisé pour éclairer des objets spécifiques dans une scène. Comme leur nom l'indique, les lumières ponctuelles ont une position définie et diffuse de la lumière dans toutes les directions à partir de cette position (Figure 2.6b). Elles sont souvent utilisées pour simuler des sources de lumière artificielles telles que des lampes, des phares de voiture ou des bougies, et peuvent également être utilisées pour éclairer des zones spécifiques dans la scène. Elles peuvent être placées à n'importe quelle position dans la scène, ce qui les rend très utiles pour éclairer des objets qui se trouvent dans des endroits difficiles d'accès pour d'autres types de lumières.



Figure 2.6 : Exemples de scènes 3D rendues avec des lumières ponctuelles.

2.3.3.2 Les lumières volumétriques

Les lumières volumétriques est une combinaison entre la lumière directionnelle et la lumière ponctuelle (Figure 2.7c). Elles permettent de simuler la façon dont la lumière se diffuse et se réfléchit dans la scène, créant ainsi des effets de lumière spectaculaires et subtils. Les lumières volumétriques sont souvent utilisées pour créer des effets de rayons de soleil, de brouillard, de brume, de nuages et d'autres phénomènes. C'est un type de lumière avancé utilisé pour créer des effets de lumière plus réalistes, elles permettent de créer des faisceaux qui peuvent être vus à travers l'air, créant ainsi des effets de diffusion et de dispersion de la lumière.

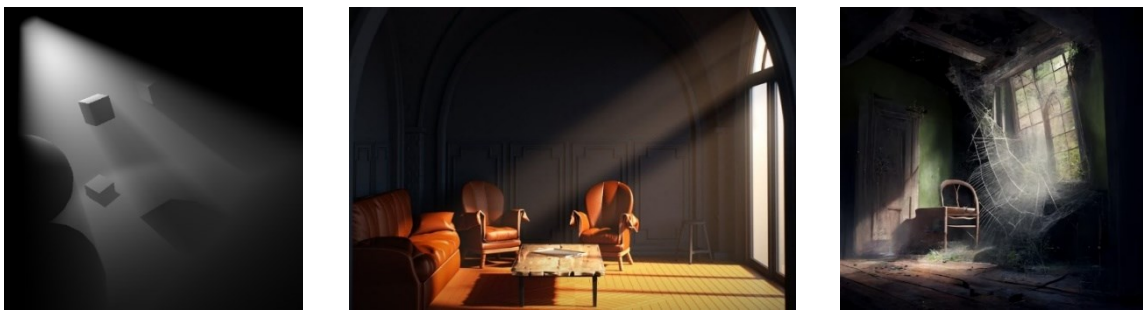


Figure 2.7 : Exemples de scènes 3D rendues avec des lumières volumétriques.

2.3.4 L'illumination

L'illumination est l'un des aspects les plus importants dans le rendu des images 3D, elle fait référence à la façon dont la lumière interagit avec les objets. Elle permet de simuler comment la lumière se reflet et se réfléchie dans une scène 3D ainsi que pour créer des ombres, des effets de transparence et des jeux de lumière. En Infographie 3D, on traite généralement deux modèles d'illumination : l'illumination locale et l'illumination globale (Figure 2.8). Par exemple, l'illumination locale peut être utilisée pour mettre en valeur un élément spécifique dans une pièce, comme une œuvre d'art ou une zone de travail, alors que l'illumination globale peut être utilisée pour fournir un éclairage uniforme dans toute la pièce. En général, une combinaison d'illumination locale et globale est utilisée pour créer un éclairage efficace et agréable.

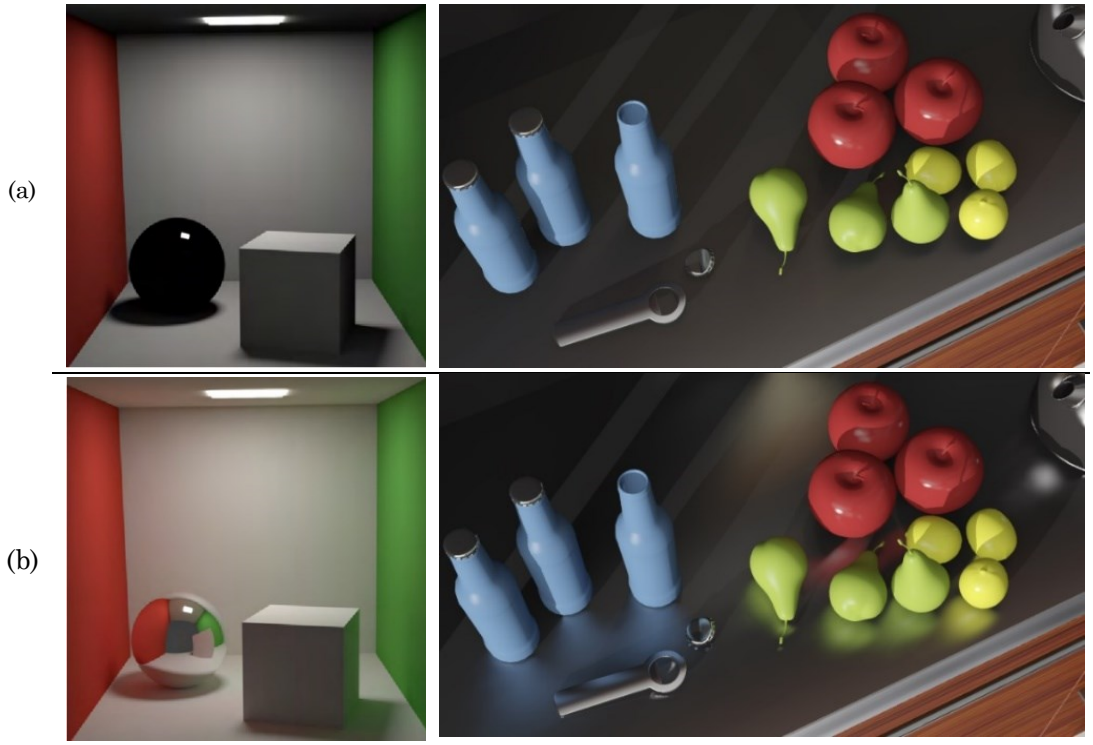


Figure 2.8 : Différence entre l'illumination locale et l'illumination globale. (a) Illumination locale. (b) Illumination globale.

2.3.4.1 Illumination locale

L'illumination locale se réfère à la façon dont la lumière est calculée et appliquée à chaque point de la surface d'un objet 3D indépendamment des autres objets et de la scène elle-même (Figure 2.8a). Contrairement à l'illumination globale, qui considère l'ensemble de la scène, l'illumination locale ne prend en compte que les sources de lumière individuelles et leur impact sur l'objet. L'illumination locale est particulièrement importante dans les jeux vidéo et les simulations en temps réel, où les performances doivent être optimisées pour assurer une expérience de jeu fluide. Celle-ci peut être calculée à l'aide de différentes techniques, telles que l'ombrage de Phong [11] ou Blinn-Phong [12], qui prennent en compte des propriétés des matériaux et de la façon dont la lumière interagit avec eux. Ces techniques permettent de créer des effets de lumière et d'ombre cohérents et garantir que l'illumination locale est calculée de manière efficace à un coup très faible.

2.3.4.2 Illumination globale

La lumière ne provient généralement pas d'une seule source lumineuse, mais de nombreuses sources dispersées tout autour de nous, même lorsqu'elles ne sont pas immédiatement visibles. L'une des propriétés de la lumière est qu'elle peut se disperser et rebondir dans de nombreuses directions, atteignant des points qui ne sont pas directement visibles. La lumière peut ainsi se refléter sur d'autres surfaces et avoir un impact indirect sur l'ombrage. Les algorithmes qui prennent cela en considération sont appelés algorithmes d'illumination globale.

L'illumination globale (Figure 2.8b) est une technique d'éclairage avancée utilisée pour simuler la façon dont la lumière se réfléchit et interagit avec les surfaces dans une scène. Contrairement à l'illumination locale, qui simule uniquement les effets des sources de lumière sur des objets individuels, l'illumination globale prend en compte les interactions de la lumière avec toutes les surfaces de la scène, y compris les surfaces réfléchissantes et les objets éclairant d'autres objets. Elle permet de traiter la façon dont la lumière se reflète et se réfléchie dans l'environnement de la scène.

Ce type d'illumination est particulièrement importante pour créer des environnements extérieurs et des scènes complexes, où la lumière peut venir de plusieurs sources et interagir avec différents types de surfaces. Cependant, le calcul de l'illumination globale peut être très coûteux en termes de temps de rendu et de puissance de traitement, ce qui peut limiter son utilisation dans les applications en temps réel comme les jeux vidéo.

L'illumination globale repose sur plusieurs techniques, les plus importantes sont :

- **Traçage de rayons (Ray-tracing)** : c'est une technique de simulation qui suit le parcours des rayons lumineux dans la scène, en calculant comment ils sont réfléchis, réfractés et diffusés par les surfaces qu'ils rencontrent [13]–[18].
- **Traçage de chemins (Path Tracing)** : c'est une technique qui utilise le principe de traçage de rayons mais d'une façon différente [19]–[24]. Contrairement au traçage de rayons qui suit les rayons de lumière de manière déterministe, le traçage de chemins utilise une approche probabiliste pour calculer les chemins de la lumière dans la scène.
- **Radiosité** : c'est une méthode de calcul basée sur la façon dont la lumière se diffuse dans une scène à partir des surfaces qui l'émettent [25]–[30]. Elle prend en compte la façon dont la lumière est réfléchie entre les surfaces, ce qui permet de simuler des effets d'ombre douce et de lumière indirecte.
- **Placage de photons (Photon Mapping)** : c'est une technique qui simule la propagation de photons dans la scène, en enregistrant leurs interactions avec les surfaces pour créer des effets de lumière indirecte [31]–[36].

Les techniques d'illumination globale offrent des résultats impressionnants, mais nécessitent souvent des ordinateurs puissants pour être rendues en temps raisonnable. Chaque technique a ses avantages et ses inconvénients, et le choix de l'une ou l'autre dépendra de la scène, des ressources informatiques disponibles et des résultats souhaités.

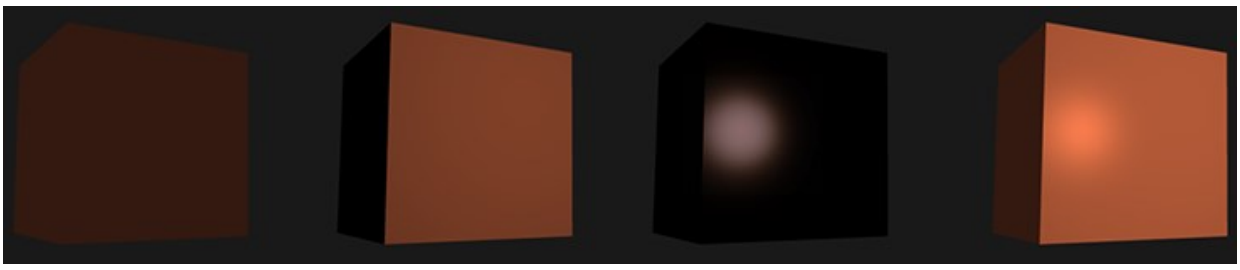
2.4 L'OMBRAGE

L'ombrage est un processus crucial pour calculer l'apparence des objets dans une scène 3D afin de simuler leurs interactions avec les différents types de lumière et pour améliorer l'apparence générale de la scène. L'ombrage rentre dans le cadre de l'illumination locale et implique l'utilisation de différents modèles mathématiques pour déterminer la couleur de chaque pixel en fonction de la lumière et de la position des objets dans la scène. Il existe plusieurs techniques d'ombrage qui peuvent être utilisées pour simuler les propriétés des surfaces des objets, chacune ayant ses propres avantages et inconvénients en ce qui concerne la qualité de rendu et le coût de calcul.

L'objectif principale d'une opération d'ombrage et le calcul de la couleur finale d'un fragment donnée en prenant en compte des caractéristiques principales comme la couleur de la source de lumière et la couleur de l'objet, l'orientation des point constituant l'objet par rapport à la source de lumière, les propriétés des matériaux qui rentre dans la constitution des surfaces des objets, etc.

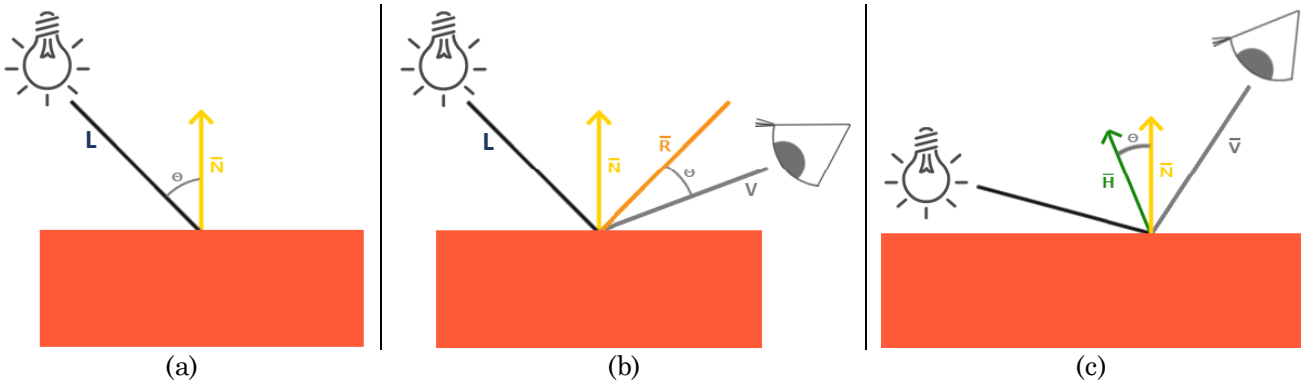
2.4.1 Les composants de l'ombrage

L'ombrage dans le monde réel est extrêmement compliqué et dépend de trop de facteurs, ce que nous ne pouvons pas nous permettre de calculer avec la puissance de traitement limitée dont nous disposons. L'ombrage en infographie 3D est donc basé sur des approximations de la réalité à l'aide de modèles simplifiés et beaucoup plus faciles à traiter. Ces modèles sont basés sur la physique de la lumière telle que nous la comprenons et se constituent de trois principaux composants qui sont : l'ombrage ambiant, l'ombrage diffus et l'ombrage spéculaire (Figure 2.9).



(a) (b) (c) (d)

Figure 2.9 : Les trois principaux composant de l'ombrage. (a) Ombrage ambiant. (b) Ombrage diffus. (c) Ombrage spéculaire. (d) Combinaison des trois composants.



(a) (b) (c)

Figure 2.10 : Cas de figures pris en charge par les modèles d'ombrage.

2.4.1.1 L'ombrage ambiant

Même lorsqu'il fait sombre, il y a généralement encore de la lumière quelque part dans le monde (la lune, une lumière lointaine), de sorte que les objets ne sont presque jamais complètement sombres (Figure 2.9a). Pour simuler cela, nous utilisons une constante qui donne toujours de la couleur à l'objet 3D. La formule mathématique de l'ombrage ambiant peut généralement être exprimée comme suit :

$$I_a = K_a \times L_a \tag{2.1}$$

Où I_a est l'intensité ambiante finale, K_a est le coefficient d'ambiance, et L_a est la couleur et l'intensité de l'éclairage ambiant.

Le coefficient d'ambiance K_a est un nombre compris entre 0 et 1 qui détermine la quantité de l'éclairage ambiant qui est appliquée à la scène. Une valeur de 0 signifie qu'aucun éclairage ambiant n'est appliqué, tandis qu'une valeur de 1 signifie que l'éclairage ambiant est appliqué à sa pleine intensité. La couleur et l'intensité de l'éclairage ambiant L_a peut être déterminé de différentes manières en fonction de la technique de rendu utilisée. Par exemple, il peut être spécifié directement dans les paramètres de l'éclairage ambiant, ou calculées en fonction de la couleur et de l'intensité des sources lumineuses de la scène.

2.4.1.2 L'ombrage diffus

L'ombrage diffus (Figure 2.9b) simule l'impact directionnel d'un objet lumineux sur un objet 3D. C'est le composant le plus significatif visuellement, plus une partie d'un objet fait face à la source lumineuse, plus elle devient lumineuse, c'est-à-dire, l'ombrage diffus donne à l'objet plus de luminosité quand ses points (pixels, fragments) sont alignés sur les rayons lumineux d'une source lumineuse. Dans la Figure 2.10a nous trouvons une source de lumière avec un rayon lumineux ciblé sur un point de l'objet. Nous devons mesurer à quel angle (l'angle θ entre le rayon lumineux $\vec{L}$ et la normal $\vec{N}$) le rayon lumineux touche l'objet dans ce point. Si le rayon lumineux est perpendiculaire à la surface de l'objet, la lumière a le plus grand impact. La formule mathématique de l'ombrage diffus est exprimée comme suit :

$$I_d = K_d \times L_d \times max(\vec{N}.\vec{L}, 0) \tag{2.2}$$

Où I_d est l'intensité diffuse finale, K_d est le coefficient de diffusion, L_d est la couleur et l'intensité de la lumière diffuse, $\vec{N}$ est la normale au point à éclairer et $\vec{L}$ est le vecteur directionnel de la lumière.

Le coefficient de diffusion K_d est un nombre compris entre 0 et 1 qui détermine la quantité de lumière diffusée par la surface de l'objet 3D. Une valeur de 0 signifie qu'aucune lumière n'est diffusée, tandis qu'une valeur de 1 signifie que toute la lumière est diffusée. La couleur et l'intensité de la source lumineuse L_d peut être déterminé de différentes manières en fonction de la technique de rendu utilisée. Par exemple, elle peut être spécifié directement dans les paramètres de la source lumineuse, ou calculées en fonction de la couleur et de l'intensité des autres sources lumineuses de la scène.

La fonction $max(\vec{N}.\vec{L}, 0)$ assure que la lumière n'est appliquée que sur les parties de la surface qui sont orientées vers la source lumineuse. Si la surface est orientée dans une

direction opposée, le produit scalaire $\vec{N}.\vec{L}$ sera négatif et la fonction $max(\vec{N}.\vec{L}, 0)$ renverra une valeur de 0, ce qui signifie que la lumière ne sera pas appliquée sur cette partie de la surface.

2.4.1.3 L'ombrage spéculaire

L'ombrage spéculaire (Figure 2.9c) simule le point lumineux d'une lumière qui apparaît sur des objets brillants. Les reflets spéculaires sont plus enclins à la couleur de la lumière qu'à la couleur de l'objet. Semblable à l'ombrage diffus, l'ombrage spéculaire est basé sur le vecteur de direction de la lumière $\vec{L}$ et les vecteurs normaux de l'objet $\vec{N}$, mais cette fois, il est également basé sur la direction de la vue $\vec{V}$ et sur les propriétés réfléchissantes des surfaces, si nous considérons la surface de l'objet comme un miroir, l'éclairage spéculaire est le plus fort partout sur l'objet 3D.

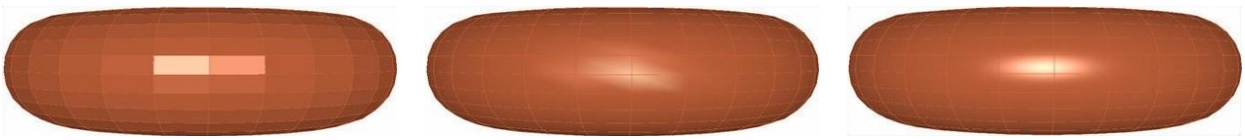
Selon le modèle d'ombrage utilisée, nous calculons un vecteur de réflexion ($\vec{R}$ dans Figure 2.10b et $\vec{H}$ dans Figure 2.10c) en réfléchissant la direction de la lumière autour du vecteur normal $\vec{N}$. Ensuite, nous calculons la distance angulaire entre ce vecteur de réflexion et le vecteur de vue. Plus l'angle entre eux est étroit, plus l'impact de la lumière spéculaire est important. La formule mathématique de l'éclairage spéculaire dépend de la technique de rendu utilisée, mais elle peut généralement être exprimée comme suit :

$$I_s = K_s \times L_s \times max(G^{\alpha}, 0) \tag{2.3}$$

Où I_s est l'intensité spéculaire finale, K_s est le coefficient de réflexion spéculaire de la surface et L_s est la couleur et l'intensité de la lumière spéculaire. La valeur G est un scalaire calculé en fonction des vecteur $\vec{V}$, $\vec{N}$ et $\vec{L}$, et α est le coefficient de brillance qui contrôle l'aspect rugueux ou lisse de la surface. Cette formule calcule l'intensité de la lumière réfléchie en fonction de la position de l'observateur, de la source de lumière et des caractéristiques de la surface,

2.4.2 Modèles d'ombrage

Un modèle d'ombrage décrit l'interaction entre la lumière et une surface en fonction de ses propriétés des matériaux ainsi que la nature de la lumière et son incidence. Les modèles d'ombrage classiques les plus couramment utilisés sont le modèle de Lambert, le modèle de Gouraud, le modèle de Phong et le modèle de Blinn-Phong (Figure 2.11).



(a) (b) (c)

Figure 2.11 : Différents modèles d'ombrage. (a) Modèle de Lambert. (b) Modèle de Gouraud. (c) Modèle de Phong.

2.4.2.1 Modèle de Lambert

Ce modèle d'ombrage suppose que la lumière est émise à partir d'une source ponctuelle et qu'elle frappe une surface plane (Figure 2.11a). Cette surface est alors éclairée de manière uniforme, indépendamment de son orientation par rapport à la source de lumière. Ce modèle est basé sur l'hypothèse que la quantité de lumière réfléchie par une surface dépend de

l'angle entre la direction de la source de lumière et la normale de la surface en ce point. Sa formule mathématique est exprimée comme suit :

$$I = I_a + I_d \tag{2.4}$$

2.4.2.2 Modèle de Phong

Le modèle d'ombrage de Phong [11] prend en compte des facteurs supplémentaires pour simuler l'éclairage sur une surface, tels que les propriétés optiques de la surface et la position de l'observateur (Figure 2.11c). Ce modèle calcule la quantité de lumière réfléchie en chaque point de la surface en utilisant une formule qui prend en compte la réflexion spéculaire, la réflexion diffuse et la réfraction de la lumière, elle est exprimée comme suit :

$$I = I_a + I_d + I_s \tag{2.5}$$

Avec :

$$G = \vec{R}.\vec{V} \tag{2.6}$$

Au début, les développeurs implémentaient le modèle de Phong dans le *Vertex Shader*. L'avantage est qu'il est beaucoup plus efficace car il y a généralement beaucoup moins de sommets par rapport aux fragments. Cependant, les couleurs finales des pixels sont les résultats de l'interpolation des couleurs des sommets constituant le maillage 3D de l'objet. Comme le montre la Figure 2.11b, le résultat était que l'éclairage n'était pas très réaliste à moins d'utiliser de grandes quantités de sommets. Lorsque le modèle de Phong est implémenté dans le *Vertex Shader*, il est appelé ombrage de Gouraud.

2.4.2.3 Modèle de Blinn-Phong

Le modèle d'ombrage de Blinn-Phong [12] est une extension du modèle d'ombrage de Phong. Il utilise une méthode de calcul basée sur les vecteurs normaux de la surface, la direction de la source de lumière, la position de l'observateur et les propriétés optiques de la surface. Tout comme le modèle d'ombrage de Phong, il calcule la quantité de lumière réfléchie en chaque point de la surface en utilisant une formule qui prend en compte la réflexion spéculaire, la réflexion diffuse et la réfraction de la lumière. Cependant, le modèle de Blinn-Phong utilise une méthode différente pour calculer la réflexion spéculaire.

Dans le modèle d'ombrage de Phong, la réflexion spéculaire est calculée en utilisant la direction de la source de lumière et la position de l'observateur. Cela peut conduire à des artefacts de rendu si la position de l'observateur ou la source de lumière sont très éloignées. Dans le modèle d'ombrage de Blinn-Phong, la réflexion spéculaire est calculée en utilisant la moitié de la direction de la source de lumière et la direction de l'observateur. Cette méthode donne des résultats plus cohérents pour la réflexion spéculaire, indépendamment de la position de l'observateur. Alors on utilise la même formule (2.5) que celle utilisée dans l'ombrage de Phong, mais le scalaire G est calculée par la formule suivante :

$$G = \vec{H}.\vec{N} \tag{2.7}$$

Le modèle d'ombrage de Blinn-Phong est souvent utilisé dans les applications graphiques de haute qualité, telles que les jeux vidéo et les films d'animation, car il permet d'obtenir des

résultats réalistes avec des coûts de calcul raisonnables. La Figure 2.12 montre une comparaison entre le modèle de Phong et celui de Blinn-Phong.

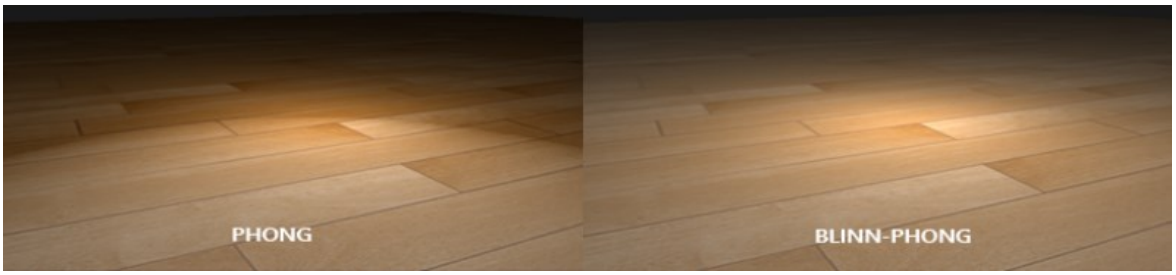


Figure 2.12 : Différence entre le modèle d'ombrage de Phong et de Blinn-Phong.

2.5 PLACAGE DE TEXTURE

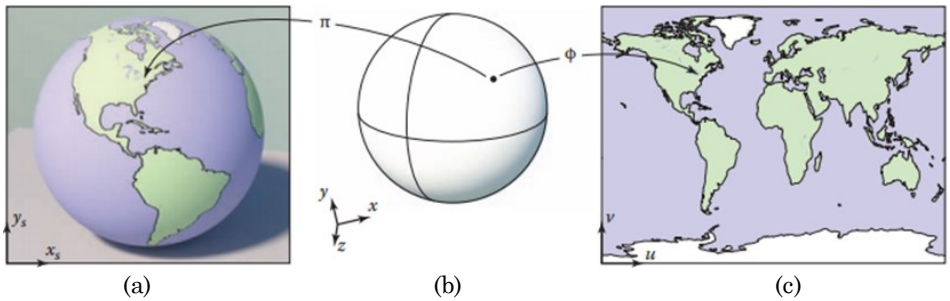
2.5.1 Principe

Le placage de texture est un processus qui consiste à appliquer une image sur la surface d'un objet 3D [10], [37]. C'est une technique importante en infographie 3D qui permet d'ajouter des détails, de la profondeur et de la réalité aux objets 3D. Le placage de texture peut être utilisé pour simuler différents matériaux tels que le bois, le métal, la pierre, le verre, etc. et pour ajouter des détails tels que des motifs, des rayures, des marques, etc. Le processus implique la création ou l'utilisation d'une texture, son application sur la surface de l'objet 3D et son ajustement pour obtenir le résultat souhaité. La texture peut être une image photographique, une peinture numérique, un motif ou un dessin créé spécifiquement pour le projet.

Le principe est relativement simple, il consiste à attribuer une paire de coordonnées 2D représentant un pixel de la texture, notée généralement (u, v), à un sommet (x, y, z) du maillage polygonal de la surface S à l'aide d'une fonction appelée paramétrisation (Figure 2.13). Ainsi, lors du rendu, les coordonnées de texture (u, v) sont interpolées en tenant compte de la projection utilisée, pour identifier le texel approprié à associer au fragment projeté à l'écran. La fonction de paramétrisation est donnée par la formule suivante :

$$F : T(u, v) \rightarrow S(x, y, z) \tag{2.8}$$

La paramétrisation $T(u, v)$ de la surface S est une fonction F qui établit une correspondance entre un ensemble $T(u, v)$ et la surface $S(x, y, z)$. Il est important de souligner que F est une fonction bijective, ce qui signifie qu'elle possède une fonction inverse appelée paramétrisation inverse.



(a) (b) (c)

Figure 2.13 : Principe du placage de texture. (a) Espace objet. (b) Surface S dans l'espace objet. (c) Espace texture.

2.5.2 Les cartes d'éclairage

Les cartes d'éclairage sont des textures qui permettent de représenter des composants qui rentre dans l'implémentation des modèles d'ombrage ou de l'illumination globale. Par exemple, dans la Figure 2.14, l'ombrage d'un plan 2D est calculé à partir d'une carte diffuse est une carte d'éclairage. Dans la Figure 2.15, deux cartes sont combinées afin de calculer l'ombrage finale des pixels d'un cube, à savoir une carte diffuse et une carte spéculaire (Figure 2.15b). Les cartes diffuse et spéculaire sont des simples textures qui représentent respectivement les couleurs des pixels qui composent l'objet 3D et ses parties brillantes.

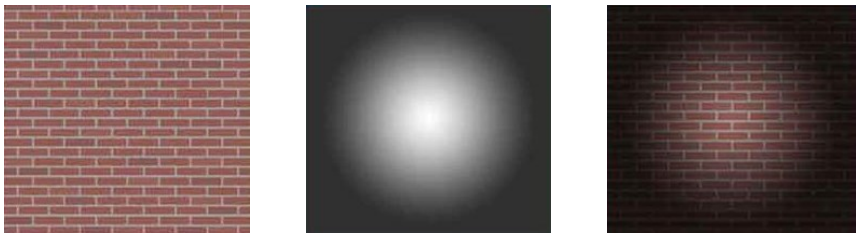


Figure 2.14 : Exemple d'ombrage en utilisant différentes cartes. (a) Carte diffuse. (b) Carte d'éclairage. (c) Diffuse × Eclairage.

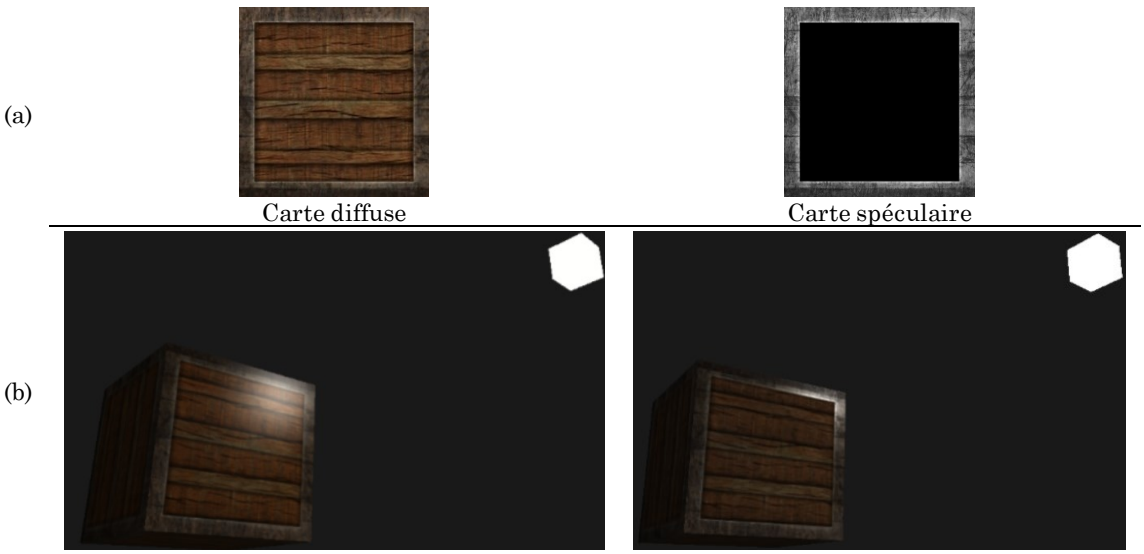
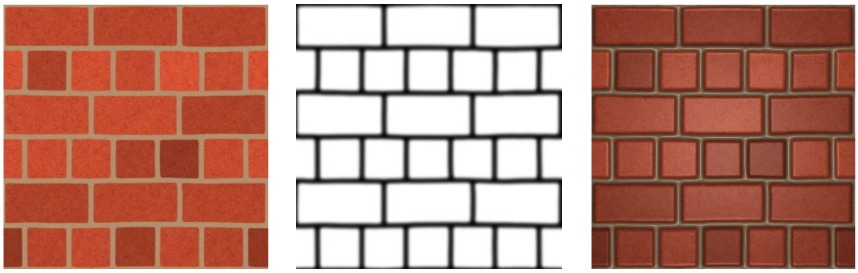


Figure 2.15 : Exemple d'ombrage. (a) Utilisation d'une carte diffuse seulement. (b) Combinaison entre la carte diffuse et la carte spéculaire.

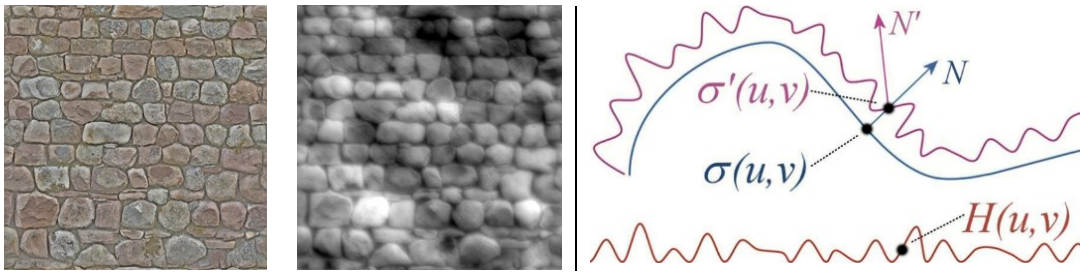
2.5.3 Placage de bosselures

Le placage de texture ne produit aucun effet de microrelief et les couleurs des pixels représentant l'objet dans la scène restent toujours les mêmes quelles que soient les conditions d'éclairage (Figure 2.16a). Pour résoudre ce problème, Blinn a introduit le placage de bosselures [38] afin de simuler des microreliefs sur les surfaces 3D sans ajouter de géométrie supplémentaire (Figure 2.16c). Le principe consiste à utiliser une texture au niveau de gris appelée carte de hauteurs ou carte de reliefs (Figure 2.16b, Figure 2.17b) qui représente les variations de hauteur de la surface de l'objet. Ces variations sont obtenues en modifiant les directions des normales à chaque pixel de l'objet 3D en fonction des valeurs de la carte de hauteurs (Figure 2.17c), ce qui permet d'avoir l'impression de relief sans ajouter de géométrie supplémentaire.



(a) (b) (c)

Figure 2.16 : Comparaison entre le placage de texture et le placage de bosselures. (a) Placage de textures. (b) Carte de hauteurs. (b) Placage de bosselures.



(a) (b) (c)

Figure 2.17 : Principe du placage de bosselures. (a) Carte diffuse. (b) Carte de hauteurs. (c) Perturbation des normales selon la carte de hauteurs H.

Le placage de bosselures n'intervient qu'au niveau de l'ombrage, qui dépend de la normale. Alors si les normales sont perturbées, cela peut donner l'impression d'un microrelief. Par conséquent, plutôt que de créer la surface σ' (Figure 2.17c), il est possible de calculer sa normale et de l'utiliser dans une formule d'ombrage.

Pour mieux comprendre le principe du placage de bosselures, on part du fait que $\sigma'(u,v)$ et le déplacement de la surface σ au point (u,v) par la grandeur $H(u,v)$, alors la formule correspondante est la suivante :

$$\sigma'(u,v) = \sigma(u,v) + H(u,v)\frac{\vec{N}(u,v)}{\|\vec{N}(u,v)\|} \tag{2.9}$$

La normale $\vec{N}'(u,v)$ de la surface modifiée $\sigma'(u,v)$ est donnée par le produit vectoriel suivant :

$$\vec{N}'(u,v) = \frac{\partial\sigma'(u,v)}{\partial u} \wedge \frac{\partial\sigma'(u,v)}{\partial v} \tag{2.10}$$

Le développement du produit vectoriel de l'équation (2.10) donne l'expression suivante pour $\vec{N}'(u,v)$:

$$\vec{N}'(u,v) = \left(\frac{\partial\sigma(u,v)}{\partial u} + \frac{\partial H(u,v)}{\partial u}\left(\frac{\vec{N}}{\|\vec{N}\|}\right)\right) \wedge \left(\frac{\partial\sigma(u,v)}{\partial v} + \frac{\partial H(u,v)}{\partial v}\left(\frac{\vec{N}}{\|\vec{N}\|}\right)\right) \tag{2.11}$$

La normale est donnée par la formule suivante :

$$\vec{N}'(u,v) = \vec{N}(u,v) + \frac{\left(\frac{\partial H(u,v)}{\partial u}\left(\vec{N} \wedge \frac{\partial\sigma(u,v)}{\partial v}\right) - \frac{\partial H(u,v)}{\partial v}\left(\vec{N} \wedge \frac{\partial\sigma(u,v)}{\partial u}\right)\right)}{\|\vec{N}\|} \tag{2.12}$$

Par conséquent, la normale perturbée est égale à :

$$\vec{N}'(u,v) = \vec{N}(u,v) + \vec{D}(u,v) \tag{2.13}$$

Avec :

$$\vec{D}(u,v) = \frac{\left(\frac{\partial H(u,v)}{\partial u}\left(\vec{N} \wedge \frac{\partial \sigma(u,v)}{\partial v}\right) - \frac{\partial H(u,v)}{\partial v}\left(\vec{N} \wedge \frac{\partial \sigma(u,v)}{\partial u}\right)\right)}{\|\vec{N}\|} \tag{2.14}$$

La normale $\overrightarrow{N'}$ sera utilisée au lieu de la normale $\vec{N}$ durant la phase de l'ombrage pour calculer la couleur de chaque pixel de l'objet 3D afin de produire un effet de déplacement et de microrelief.

Le placage de bosselures est une technique efficace pour ajouter des détails de surface sans ajouter de géométrie supplémentaire tout en économisant les ressources matérielles de l'ordinateur. Cependant, cette technique est limitée et ne peut pas remplacer complètement la géométrie réelle pour créer des objets détaillés.



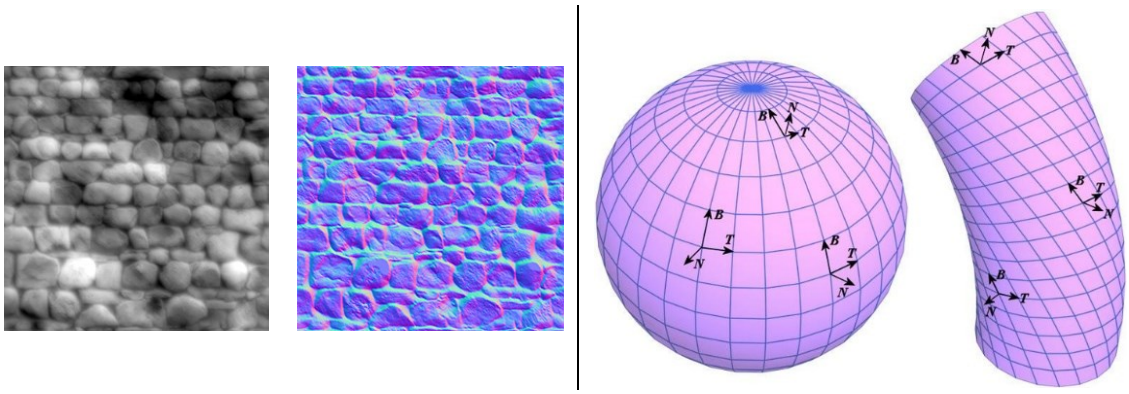
(a) (b)

Figure 2.18 : Comparaison de rendu entre le placage de texture (a) et le placage de bosselures (b).

2.5.4 Placage de normales

Le placage de bosselures repose sur le calcul des dérivées partielles de la carte de hauteurs pour chaque pixel constituant l'objet 3D et de les utiliser dans la formule (2.12) pour calculer la normale perturbée. Ce calcule est effectuée durant la phase de rendu et peut ralentir sa vitesse dans le cas des scènes complexes. Pour pallier à ce problème, le placage de normales a été introduit dans [39].

La technique du placage de normale est une variante du placage de bosselure qui repose sur une texture calculée à partir de la carte de hauteurs appelée carte de normales est sur un espace local associe à chaque point (u,v) de la surface σ appelé espace tangent.



(a) (b) (c)

Figure 2.19 : Les éléments fondamentales du placage de normales. (a) Carte de hauteurs. (b) Carte de normales. (c) Espace tangent.

2.5.4.1 La carte de normales

L'idée principales du placage de normales est de calculer les normales perturbées indépendamment de la surface σ et de les stocker dans une texture RGB appelée carte de normales (Figure 2.19b) dont les canaux r , g et b représentant respectivement les composants x, y et z de la normale. La carte de normales est calculée à partir des dérivées partielles de la carte de hauteurs H (Figure 2.19a) en utilisant par exemple la convolution par un filtre gradient. Les normales calculées sont exprimées dans l'espace texture et peuvent être appliquée à n'importe quelle surface 3D. Cela permet d'éviter le calcul des dérivées partielles de la carte de hauteurs H pendant la phase de rendu par la relation (2.12) qui consomme beaucoup de ressources.

En appliquant la carte de normales sur la surface de l'objet, les normales de chaque pixel seront perturbées en fonction de la direction des normales de la texture, ce qui donne l'impression de reliefs et de petits déplacements et avec un coup relativement faible.

2.5.4.2 L'espace tangent

L'espace tangent est un espace associe à chaque point (u, v) de la surface σ (espace associe à chaque sommet du maillage 3D) qui est représenté par le vecteur tangent $\vec{T}$, le vecteur binormal $\vec{B}$ et le vecteur normal $\vec{N}$ (Figure 2.19c). Ces vecteurs sont calculés selon les relations suivantes :

$$\vec{T} = \frac{\partial \sigma(u,v)}{\partial u}, \qquad \vec{B} = \frac{\partial \sigma(u,v)}{\partial v}, \qquad \vec{N} = \vec{T} \wedge \vec{B} \tag{2.15}$$

Les vecteurs $\vec{T}$ et $\vec{B}$ sont des vecteurs tangents à la surface σ est sont appelées respectivement tangente et binormale et forment avec la normale $\vec{N}$ un espace local en chaque point (u, v) de σ (espace locale associé à chaque sommet du maillage 3D) appelé espace tangent ou espace TBN.

Dans la relation (2.12), la normale $\overrightarrow{N'}$ est exprimée en fonction des dérivées partielles de H et des dérivées partielles de σ, alors pour avoir un ombrage correct et relativement propre à la surface σ de l'objet 3D, cette relation doit être restituée en faisant un changement d'espace entre l'espace de texture vers l'espace TBN.

Notons par $\vec{N}_H$ la normale extraite à partir de la carte de normales au point (u, v), le calcul de la normale $\overrightarrow{N'}$ est donné par la formule suivante :

$$\overrightarrow{N'} = TBN \,.\, N_H(u,v) \tag{2.16}$$

Avec :

$$TBN = \begin{bmatrix} T_x & B_x & N_x \\ T_y & B_y & N_y \\ T_z & B_z & N_z \end{bmatrix} \tag{2.17}$$

Les vecteurs *lumière* et *vue* sont généralement exprimées dans l'espace global de la scène, il faudra alors les transformés aussi dans l'espace tangent avant de les utiliser dans les formules d'ombrage.

Les trois composantes de l'espace tangent sont calculées directement à partir de la surface paramétrique σ, lorsque celle-ci est disponible (plan, cylindre, sphère, cône, tore, etc.) en utilisant les relations (2.15). Mais dans le cas d'un maillage polygonal quelconque, il faut passer par les coordonnées géométriques (x, y, z) et les coordonnées de texture (u, v) des sommets constituant le maillage triangulaire [40].

2.6 PLACAGE DE DEPLACEMENT PAR SOMMET

Contrairement au placage de bosselures qui se content juste de produire des illusions de petits déplacements afin de produire un effet de microrelief, le placage de déplacement par sommet [41], [42] produit des déplacements réel de la surface de l'objet 3D. Cependant, il peut être coûteux en termes de performance, car il nécessite un grand nombre de sommets pour créer des détails de relief précis. Il est donc important d'optimiser la géométrie du maillage avant d'appliquer cette technique, pour minimiser l'impact sur les performances lors du rendu. Le placage de déplacement par sommet repose sur le déplacement des sommets et sur la densification du maillage 3D (Figure 2.20).

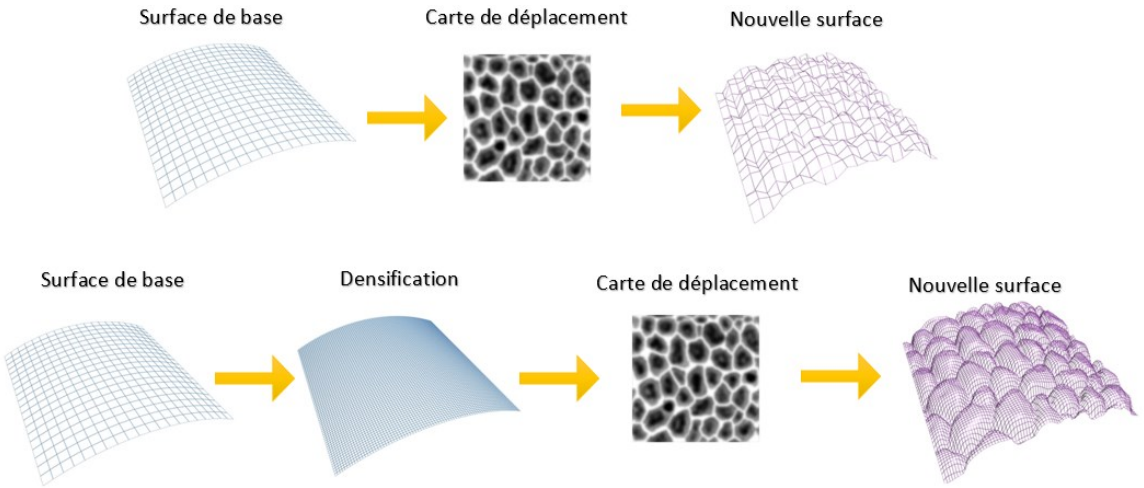
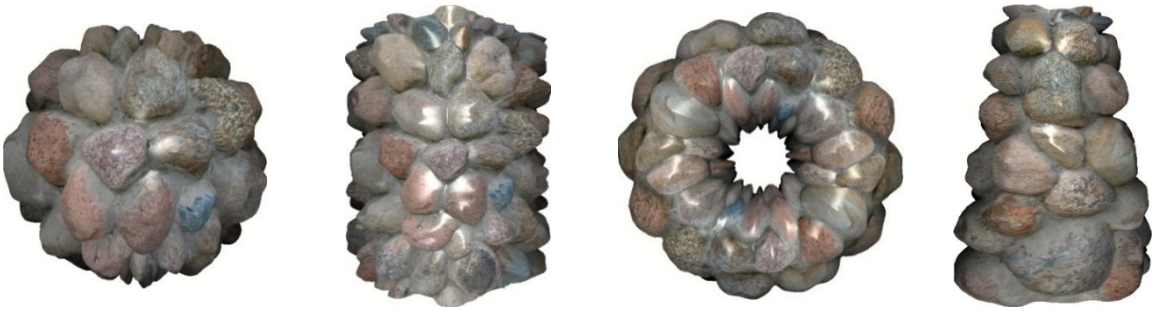


Figure 2.20 : Principe du placage de déplacement par sommet.

2.6.1 Déplacement des sommets

Le placage de déplacement par sommet [41], [42] utilise la carte de hauteurs d'une autre manière par rapport au placage de bosselures. Elle consiste à déplacer les sommets de la surface suivant la normale et en fonction des valeurs stockées dans la carte de hauteurs. Les parties les plus claires de la carte de hauteurs seront déplacées vers le haut, tandis que les parties les plus sombres seront déplacées vers le bas (dans le cas d'une carte de profondeurs, ce mécanisme est inversé). En effet, selon la Figure 2.17, la surface σ sera réellement déplacée en chaque point (u, v) selon la normale et selon une valeur stockée dans H au coordonnées (u, v) en utilisant la formule (2.9). Une fois qu'un sommet du maillage a été déplacé, il est nécessaire de définir sa normale. Pour les maillages arbitraires, cette normale est calculée en prenant la moyenne des normales des primitives adjacentes au sommet. Toutefois, dans le cas du placage de déplacement, la formule (2.12) issue du placage de bosselures permet de calculer directement la normale.

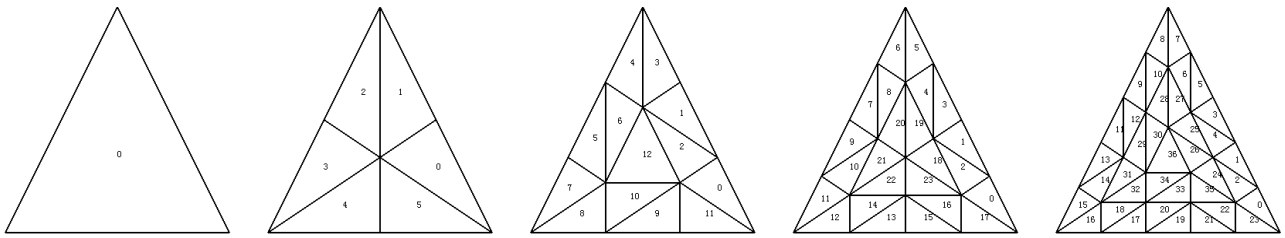


Sphere Cilyndre Tore Cône

Figure 2.21 : Placage de déplacement par sommet appliqué sur différents objets 3D.

2.6.2 Densification du maillage 3D

La densification du maillage 3D est le processus d'ajout de nouvelles primitives (sommets et polygones) pour augmenter sa résolution et obtenir un maillage plus détaillé (Figure 2.20, Figure 2.22). Cette technique est couramment utilisée dans la modélisation 3D pour améliorer la qualité visuelle des modèles, ajouter des détails fins, ou adapter un maillage à des besoins spécifiques, tels que la simulation physique ou l'animation.



Niveau 1 Niveau 2 Niveau 3 Niveau 4 Niveau 5

Figure 2.22 : Exemple de processus de tessellation par le GPU.

Dans le cadre du placage de déplacement par sommet, la densification du maillage est réalisée par un processus appelé subdivision de surface ou tessellation. C'est l'une des méthodes les plus couramment utilisées pour densifier un maillage 3D. Elle est devenue une partie intégrante au niveau du pipeline graphique moderne à travers des unités programmables qui sont le *Tessellation Shader* et le *Geometry Shader*. En effet, grâce aux architectures parallèles des cartes graphiques, la vitesse et le temps du processus de densification sont largement optimisés en exploitant leur capacité à effectuer de nombreuses opérations simultanément.

La tessellation est une fonctionnalité qui permet de subdiviser dynamiquement la géométrie d'un maillage pendant le rendu en temps réel en subdivisant chaque polygone en sous polygones et par conséquent l'ajout de nouveaux sommets (Figure 2.22). Elle peut être utilisée pour augmenter la densité du maillage 3D de manière adaptative en fonction de la distance de la caméra, du niveau de détail souhaité ou d'autres critères, offrant ainsi un moyen efficace de densifier le maillage en fonction des besoins de rendu.

Il est important de noter que la densification du maillage 3D peut augmenter la complexité du modèle et peut également avoir un impact sur les performances de rendu en raison de la charge de calcul supplémentaire associée à un maillage plus détaillé. Il est donc nécessaire de prendre en compte les exigences du projet, les performances du système et les

contraintes de temps pour choisir la méthode de densification la mieux adaptée à votre besoin spécifique. Le placage de déplacement adaptatif, décrit dans [43], [44] permet de moduler la densification (ou tessellation) du maillage en fonction des variations de la carte de hauteurs, permettant ainsi de réduire considérablement le nombre de polygones.

2.7 PLACAGE DE DEPLACEMENT PAR PIXEL

Le placage de déplacement par pixel [45] est une technique basée sur le placage de texture [10], [37]. Il s'inspire à la fois du placage de bosselures [38], [39] qui procède au niveau des pixels de la texture des microreliefs, et du placage de déplacement par sommet [41], [42] qui procède au niveau des sommet du maillage 3D. L'objectif est d'obtenir un rendu similaire à celui du placage de déplacement par sommet, tout en évitant d'augmenter la densité du maillage de base, et en préservant la qualité visuelle globale de la scène.

La Figure 2.23 montre une comparaison de différentes techniques de placage de texture. Comme le montre la figure, la densité de maillage est la même dans les images (a), (b) et (d), mais le placage de déplacement par pixel permet de produire une géométrie très détaillée avec un maillage de faible densité.

Plus de l'espace tangent (Figure 2.19c), le placage de déplacement par pixel est basé sur deux autres éléments principaux : la carte de déplacement et le traçage de rayons.

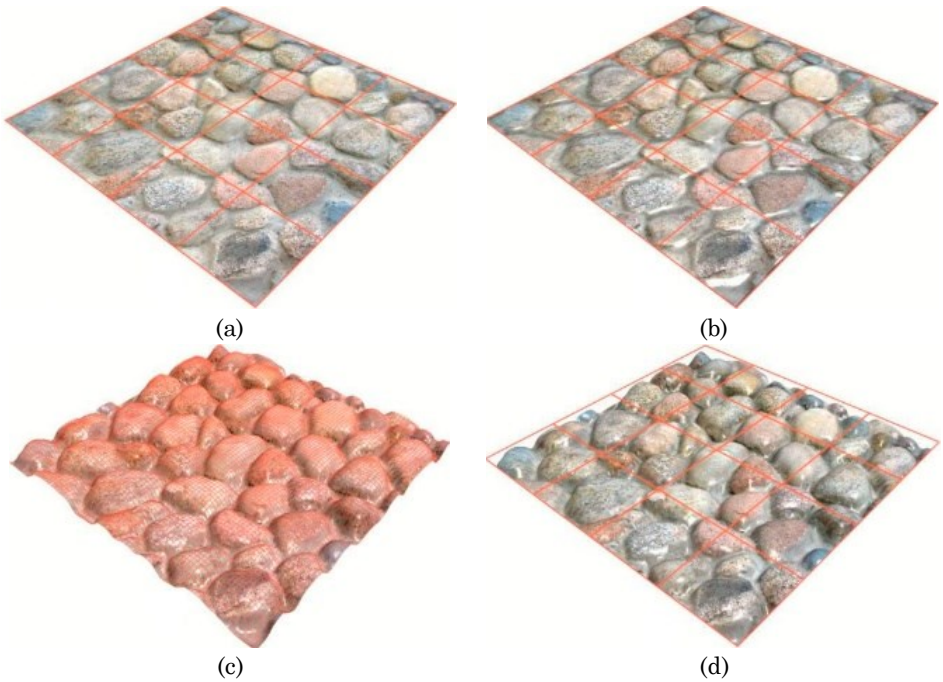
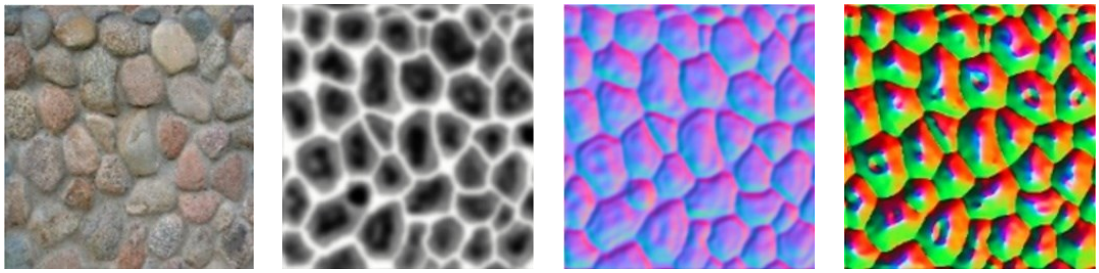


Figure 2.23 : Différentes techniques du placage de texture [46]. (a) Placage de textures. (b) Placage de bosselures. (c) Placage de déplacement par sommet. (d) Placage de déplacement par pixel.

2.7.1 La carte de déplacement

La carte de déplacement (Figure 2.24d) est une texture RGBA dont les pixels ne sont pas utilisés pour stocker des couleurs, mais des données géométriques (profondeurs et normales). Dans le canal α, on stocke les profondeurs associées au microrelief plaqué sur la surface 3D.

Les trois autres canaux : rouge, vert et bleu, sont utilisés pour stocker les trois composantes x, y et z de la normale, qui sont calculées à partir des profondeurs. Comme la composante z de la normale peut être récupérée en fonction des deux autres, le canal bleu peut être libéré pour stocker d'autres données utilisées par certaines techniques comme le traçage de cônes. Dans ce cas, la carte de déplacement peut être nommée selon cette technique (c'est-à-dire la carte de cônes).



(a) (b) (c) (d)

Figure 2.24 : (a) Texture (carte diffuse). (b) Carte de profondeurs. (c) Carte de normales. (d) Carte de cônes.

2.7.2 Traçage de rayons

Le traçage de rayons est un algorithme qui recherche l'intersection du rayon de vue et du microrelief stocké dans la carte de déplacement (Figure 2.25). Cette recherche est effectuée dans l'espace texture pour chaque pixel issu du maillage 3D. Le principal problème du placage de déplacement par pixel est de trouver le premier point d'intersection.

Comme le montre la Figure 2.25, la première intersection est représentée par le point (x, y) le long du rayon de vue. Ainsi, pour avoir une illusion de déplacement de relief, le fragment (u, v) sera texturé à l'aide du texel (x, y) au lieu du texel (u, v). La contrainte de la vitesse ne permet pas une recherche exacte, ce qui oblige à trouver un point le plus proche possible du point d'intersection. Les différentes approches pour effectuer cette recherche sont présentées dans [47]. La Figure 2.26 montre des objets rendus avec la technique du placage de déplacement par pixel.

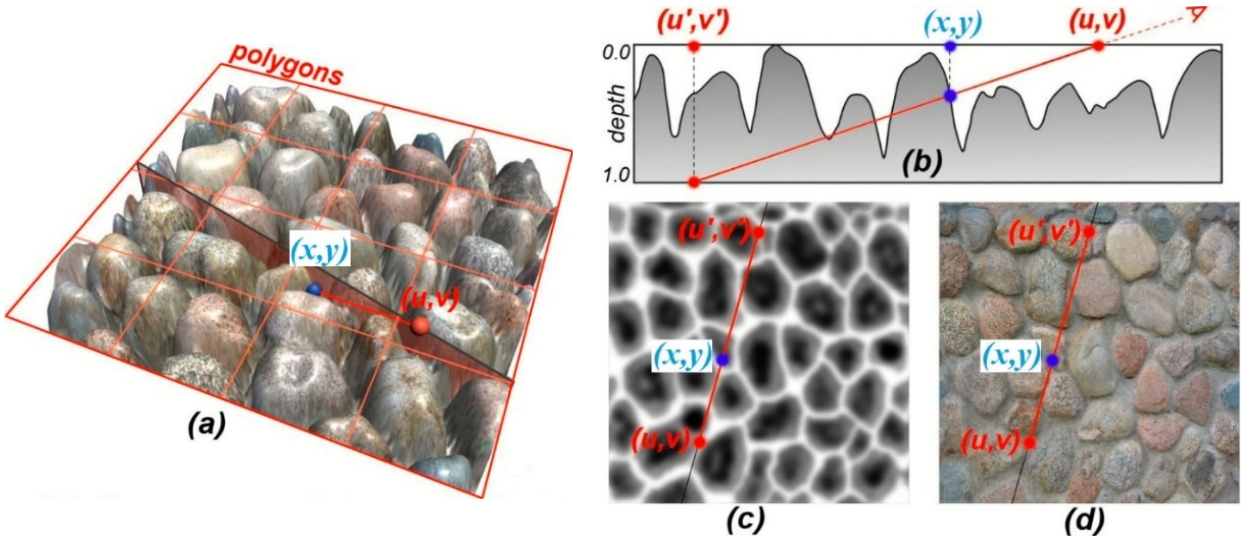


Figure 2.25 : Traçage de rayons dans la carte de déplacement [46]. (a) Vue 3D du traçage de rayons. (b) La tranche du relief incluant le rayon de vue. (c) La carte de profondeurs. (d) La texture correspondante.

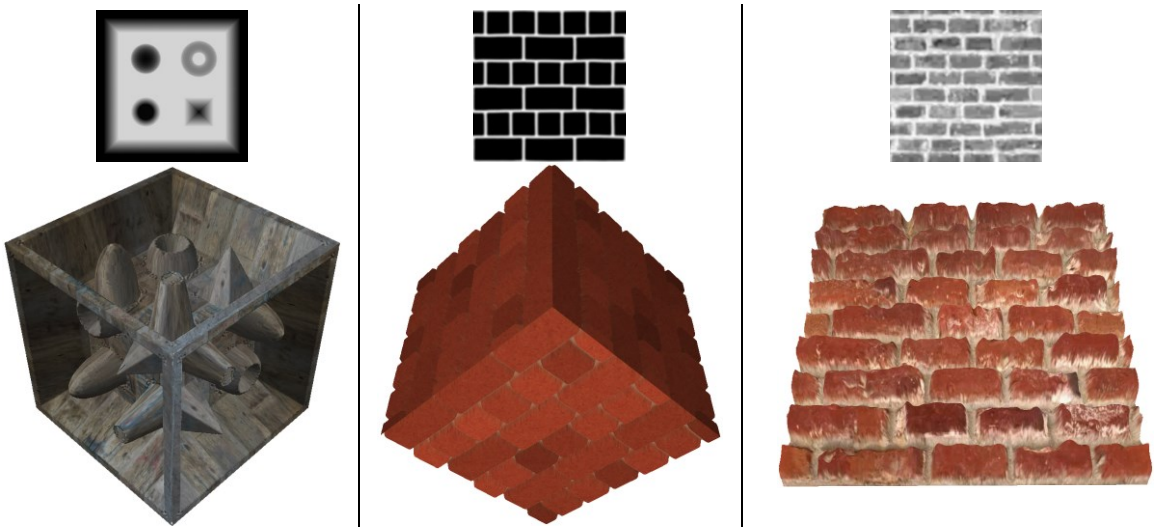


Figure 2.26 : Exemples d'objets 3D rendus avec le placage de déplacement par pixel.

2.8 CONCLUSION

Les modèles d'ombrage et le placage de texture sont des éléments clés de l'infographie 3D. En combinant ces techniques, il est possible de créer des images et des environnements virtuels qui sont visuellement attrayants. Ces techniques sont largement utilisées dans les industries du jeu vidéo, du cinéma, de la publicité et de la simulation, où la qualité visuelle est primordiale.

Les modèles d'ombrage sont utilisés pour simuler la façon dont la lumière interagit avec les objets 3D. Les différents modèles d'ombrage offrent des avantages et des limites au niveau de la qualité visuelle et du temps de traitement, et le choix du modèle à utiliser dépendra des besoins spécifiques du projet. Le placage de texture est également important pour créer des détails visuels sur les objets 3D, car les textures peuvent être utilisées pour ajouter des motifs, des couleurs et des détails sur les surfaces, ce qui peut améliorer la qualité visuelle des scènes 3D.

Les modèles d'ombrage et le placage de texture sont également importants dans le cadre de la réalité virtuelle et la réalité augmentée. Dans ces deux contextes, les images et les environnements doivent être rendus en temps réel, ce qui signifie que les modèles d'ombrage et les textures doivent être conçus de manière à être traités efficacement par les processeurs graphiques modernes. Les techniques d'optimisation, telles que l'utilisation de textures à faible résolution ou l'utilisation de modèles d'ombrage simplifiés, sont donc également importantes pour garantir des performances optimales et une expérience utilisateur fluide.

Après avoir bien assimilé les principes et les concepts des modèles d'ombrage ainsi que les techniques de placage de texture, notamment la technique de placage de déplacement par pixel, il est temps pour aborder en détail nos contributions proposées dans cette thèse. Avant cela, nous allons d'abord recenser les différents travaux de recherche antérieurs relevant de ces contextes ainsi que leurs problématiques associées. Cela nous permettra d'avoir une vue d'ensemble des techniques de placage de déplacement par pixel. Un tel sujet sera l'objectif du chapitre suivant.

Chapitre 3

Les travaux realises en : Techniques de placage de deplacement par pixel

3.1 INTRODUCTION

Le placage de déplacement par pixels est une technique de rendu 3D qui permet d'appliquer des déplacements à une surface sans augmenter la complexité du modèle 3D lui-même. Cette technique fonctionne en utilisant des textures spéciales appelées *cartes de déplacement*. Ces textures stockent des informations de déplacement sous forme de valeurs de profondeurs ou de hauteurs pour chaque pixel de la texture. Lors du rendu, ces informations sont utilisées par les algorithmes de traçage de rayons pour calculer la couleur de chaque pixel constituant l'objet 3D. Cela permet de simuler des variations de profondeur sur la surface sans modifier la géométrie sous-jacente du modèle 3D.

Le traçage de rayons est utilisé dans de nombreux domaines de l'infographie 3D, incluant la simulation de la lumière, la gestion des collisions, la modélisation physique, la réalité virtuelle, les jeux vidéo, les films d'animation et bien d'autres. Il peut être implémenté de différentes manières en utilisant des algorithmes de recherche et des techniques spécifiques pour optimiser les performances et obtenir des résultats précis et efficaces. Généralement, il consiste à émettre un rayon depuis un point d'origine dans l'espace (par exemple à partir de la caméra ou d'une source lumineuse) dans une certaine direction et de le tester pour déterminer s'il intersecte un objet dans un espace appartenant à la scène 3D. Si une intersection est trouvée, es informations associées telles que la position de l'intersection, la normale à la surface, la couleur, ou les propriétés de l'objet peuvent être exploitées pour calculer différents effets visuels.

Dans le cadre du placage de déplacement par pixel, le traçage de rayons consiste à effectuer des recherches dans la carte de déplacement afin de déterminer l'intersection du rayon de vue avec les microreliefs qui sont représentés par des profondeurs dans cette carte. Cette recherche se déroule en plusieurs étapes :

- **Génération du rayon de vue** : à partir d'une caméra virtuelle dans la scène 3D, un rayon est généré en fonction de la position du pixel de l'image sur lequel on souhaite effectuer le traçage de rayons.
- **Intersection du rayon du vue avec le microrelief** : le rayon de vue est ensuite testé pour déterminer s'il intersecte le microrelief. Ceci est généralement réalisé en effectuant des tests de comparaison entre la profondeur courant le long du rayon de vue et les profondeurs stockées dans la carte de déplacement en utilisant des algorithmes d'intersection itératifs.
- **Utilisation de la carte de déplacement** : si une intersection est trouvée, les coordonnées (x, y) du point d'intersection est utilisé pour déterminer des informations sur la normal ou le déplacement de la surface à cette position à partir de la carte de déplacement.
- **Application du microrelief** : les informations de normales, de profondeurs ou de déplacement extraites de la carte de déplacement sont utilisées pour déplacer la position d'intersection du rayon, en ajoutant les détails de relief sans modifier la géométrie de l'objet 3D.

- **Calcul des effets visuels** : une fois que l'effet de microrelief a été appliqué au point d'intersection, les informations liées à celui-ci peuvent être utilisées pour calculer d'autres effets visuels tels que l'ombrage, les ombres, la réflexion, la réfraction, etc., en fonction de la nouvelle géométrie résultante.

Une meilleure recherche permet d'avoir des objets 3D les plus réaliste possible et contribue à l'amélioration de la qualité visuelle de la scène. Cependant, cette recherche peut également être coûteuse en termes de temps de calcul, car elle nécessite d'effectuer des accès et des interpolations dans la carte de déplacement, ce qui peut être optimisé en utilisant des techniques d'accélération et de parallélisme sur les GPU modernes pour obtenir des performances en temps réel.

Le placage de déplacement par pixels peut être réalisé à différents niveaux de détail, allant de simples détails subtils à des reliefs plus complexes et détaillés. Il offre plusieurs avantages par rapport au placage de déplacement par sommet, notamment :

- **Efficacité en termes de performance** : le placage de déplacement par pixels ne modifie pas la géométrie sous-jacente de l'objet 3D, ce qui signifie qu'il n'ajoute pas de sommets supplémentaires au maillage. Cela permet de préserver la complexité de l'objet 3D et d'éviter les coûts de performance associés à la manipulation de géométrie supplémentaire.
- **Flexibilité artistique** : le placage de déplacement par pixels permet d'utiliser des cartes de déplacement pour créer des détails géométriques sur une surface 3D, ce qui offre une grande flexibilité artistique. Les cartes de déplacement peuvent être créées et éditées de manière indépendante des objets 3D, ce qui permet aux artistes de modifier les détails géométriques sans avoir à modifier la topologie de l'objet 3D lui-même.
- **Précision des détails** : le placage de déplacement par pixels peut offrir une plus grande précision des détails géométriques par rapport au placage de déplacement par sommet. En utilisant des cartes de déplacement de haute résolution, il est possible d'obtenir des détails géométriques très fins et complexes sur la surface 3D, ce qui peut donner des résultats visuellement convaincants.

Dans ce chapitre, nous allons évoquer les principales techniques du placage de déplacement par pixel et les problématiques liées, à savoir, la recherche du point d'intersection du rayon de vue avec le microrelief, le traitement de la silhouette et la pris en charge de l'auto-ombrage.

3.2 RECHERCHE DU POINT D'INTERSECTION

Le principe de base des techniques de placage de déplacement par pixel est de déplacer chaque pixel d'une texture 2D en fonction de sa position sur la surface à laquelle il est appliqué. Ce déplacement est contrôlé par la carte de déplacement, qui utilise des valeurs de profondeurs (ou de hauteurs) pour déterminer la quantité de déplacement à appliquer à

chaque pixel. Généralement, ils existent deux familles de techniques, les techniques sans encodage de l'espace vide et les techniques avec encodage de l'espace vide.

3.2.1 Techniques sans encodage de l'espace vide

Ces techniques reposent sur des recherches aveugle, c'est à dire la recherche de l'intersection du rayon de vue avec le microrelief est effectuée sans aucune information préalable concernant les voisinages des pixels de la carte de profondeurs.

3.2.1.1 Placage de parallaxe

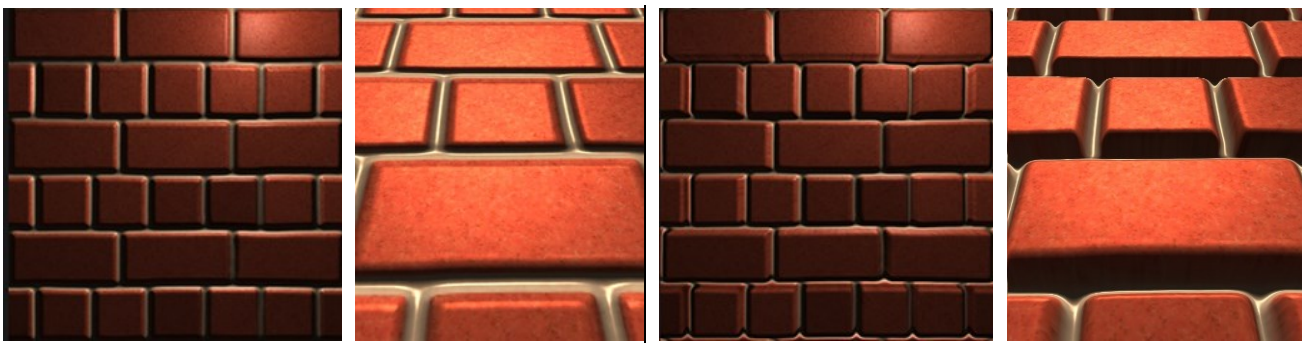
Le placage de parallaxe est une extension du placage de bosselures qui permet de simuler une certaine profondeur sur des textures appliquées aux surfaces des objets 3D. Cette technique est souvent utilisée pour créer des effets de relief sur des éléments de décor, tels que des murs, des sols ou des panneaux.

Le principe de base du placage de parallaxe est de créer une illusion de profondeur en déplaçant les pixels de la texture selon l'angle de vue de la caméra. Lorsque l'on regarde la texture de face, les pixels sont alignés les uns sur les autres et la surface semble plate. Mais lorsqu'on change l'angle de vue, les pixels sont déplacés de manière différente en fonction de leur position sur la texture, créant ainsi l'illusion de profondeur (Figure 3.1).

Il existe plusieurs méthodes pour réaliser le placage de parallaxe (Figure 3.2), chacune ayant ses avantages et ses inconvénients en termes de qualité d'image et de performance. Certaines techniques sont basées sur des calculs complexes de déplacement des pixels, tandis que d'autres sont plus simples et utilisent des effets spécifiques des logiciels et applications 3D. Dans la Figure 3.2, les courbes en bleu représentent les valeurs de la carte de profondeurs alors que le vecteur $\vec{V}$ représente le rayon de vue.

Les méthodes les plus importantes du placage de parallaxe sont :

- Placage de parallaxe simple.
- Placage de parallaxe avec limitation du décalage.
- Placage de parallaxe en tenant compte de la pente.
- Placage de parallaxe itérative.



(a) (b)

Figure 3.1 : Différence entre le placage de normales (a) et le placage de parallaxe (b).

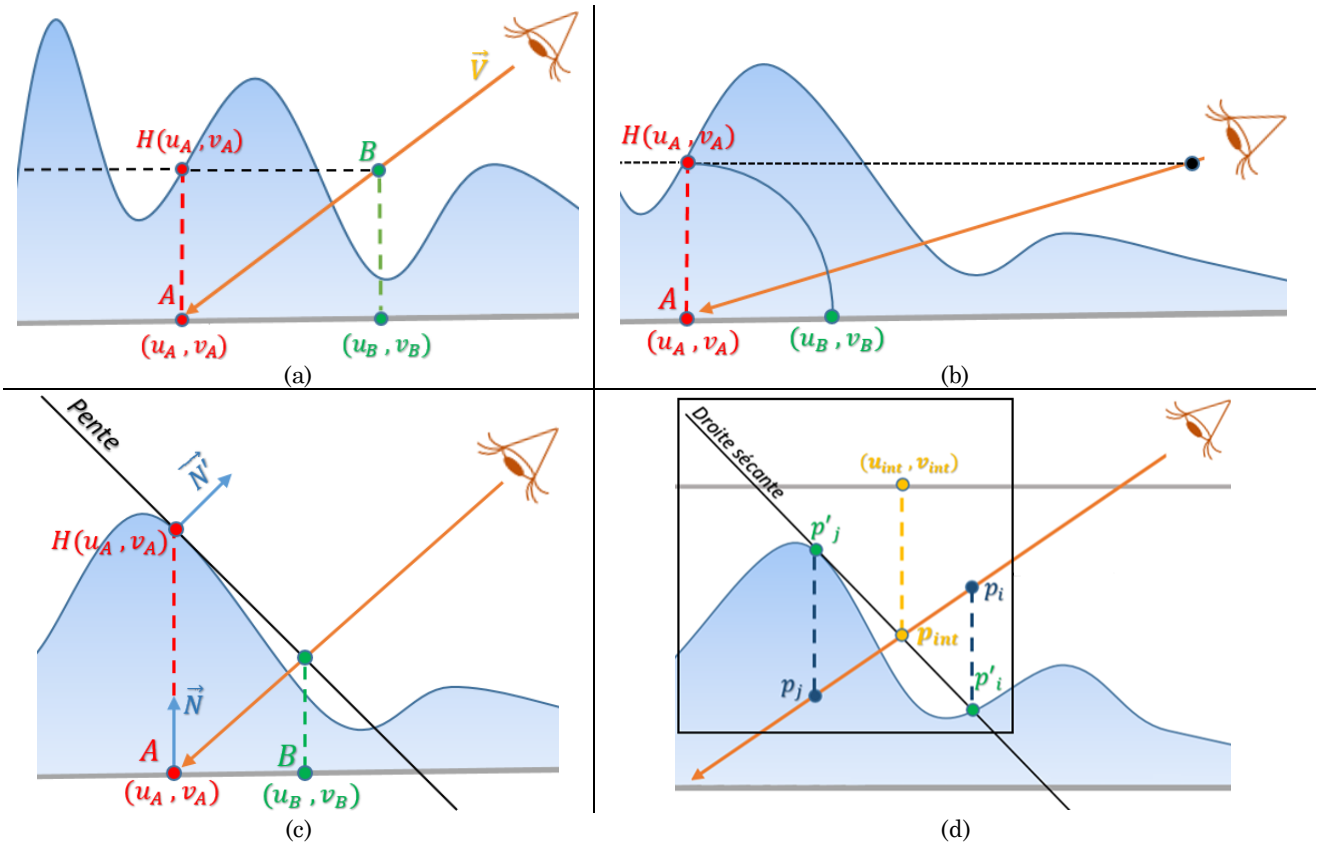


Figure 3.2 : Différentes méthodes du placage de parallaxe pour la recherche du point d'intersection [48].

A. Placage de parallaxe simple

Le placage de parallaxe simple [49] effectue une recherche approchée de l'intersection entre le rayon de vue et le relief stocké dans la carte de déplacement (Figure 3.2a). Ce point est défini par l'intersection du rayon de vue et la ligne horizontale qui passe par la profondeur du relief au point courant ajoutant ainsi l'effet de parallaxe. Cette approche permet de simuler l'apparence de relief sur une surface d'une manière très simples. Cependant, son approche approximative, qui suppose que la profondeur est uniforme autour du point (u, v), limite son utilisation à des microreliefs irréguliers.

Le placage de parallaxe simple repose sur le décalage des coordonnées de texture (u_A, v_A) en utilisant le vecteur de vue $\vec{V}$ et la valeur $H(u_A, v_A)$ obtenue à partir de la carte de profondeurs. Ce décalage est basé sur l'hypothèse que les valeurs de profondeur sont régulières autour du point (u_A, v_A), ce qui permet de calculer les nouvelles coordonnées de texture pour obtenir l'effet de parallaxe. Les nouvelles coordonnées de textures (u_B, v_B) sont calculées par la relation suivante :

$$(u_B, v_B) = (u_A, v_A) - H(u_A, v_A)\left(\frac{V_x}{V_z}, \frac{V_y}{V_z}\right) \tag{3.1}$$

Bien que le placage de parallaxe soit une méthode qui peut améliorer la qualité de rendu des objets 3D constitués de polygones complexes, il présente certaines limitations. Par exemple, il ne peut pas simuler l'occlusion, l'auto-ombrage ou traiter les silhouettes. De plus, dans sa forme simple, le placage de parallaxe peut présenter un défaut majeur lorsque l'angle de vue devient plus rasant, c'est à dire que les valeurs de décalage peuvent approcher l'infini, ce qui donne un résultat qui semble aléatoire.

B. Placage de parallaxe avec limitation du décalage

Le placage de parallaxe avec limitation du décalage [50] est une variante du placage de parallaxe qui vise à résoudre le problème des valeurs approchant l'infini lorsque l'angle de vue devient plus rasant (Figure 3.2b). Cette technique implique de limiter la valeur du décalage maximal appliqué aux coordonnées de texture afin d'éviter des résultats non cohérents.

Une approche courante pour limiter le décalage est d'utiliser une fonction de seuillage pour restreindre la valeur du décalage à un certain seuil. Par exemple, une fonction linéaire ou une fonction exponentielle peut être utilisée pour réduire le décalage à mesure que l'angle de vue s'approche de la vue de dessus. Cette approche permet de contrôler les effets de parallaxe en fonction de l'angle de vue, afin de maintenir un rendu correct même dans des conditions de visualisation difficiles. Cependant, il est important de noter que chaque implémentation de cette technique peut avoir des paramètres et des résultats spécifiques, et peut nécessiter des ajustements pour obtenir les effets de relief souhaités, et cela en fonction des besoins spécifiques d'un projet ou d'une application.

La limitation du décalage peut être mise en œuvre en supprimant la division par V_z de l'équation (3.1), ce qui rend la mise en œuvre encore plus simple :

$$(u_B , v_B) \approx (u_A , v_A) - H(u_A , v_A)\big(V_x, V_y\big) \tag{3.2}$$

C. Placage de parallaxe en tenant compte de la pente

Le placage de parallaxe en tenant compte de la pente [51] est une approche plus avancée du placage de parallaxe qui prend en considération la pente de la surface de l'objet 3D pour calculer le décalage de texture (Figure 3.2c). Contrairement au placage de parallaxe simple qui suppose que la profondeur est uniforme autour d'un point de texture, le placage de parallaxe avec pris en compte de la pente considère que la profondeur varie en fonction de la pente de la surface. Dans cette approche, la pente est généralement calculée à partir de la normale $\overrightarrow{N'}$ qui représente la direction perpendiculaire à la surface à un point donné. La normale $\overrightarrow{N'}$ est utilisée pour déterminer l'angle d'incidence de la vue et la direction de la réfraction de la lumière à travers la surface. Ces informations sont ensuite utilisées pour ajuster le décalage des coordonnées de texture en utilisant la relation suivante :

$$(u_B, v_B) \approx (u_A, v_A) + H(u_A, v_A)(V_x, V_y){N'}_z \tag{3.3}$$

L'utilisation de la pente dans le calcul du décalage permet d'obtenir des résultats plus précis, en particulier dans les zones de forte inclinaison où la profondeur varie significativement. Cela permet de mieux simuler les effets de relief et d'auto-ombrage, et de produire un rendu plus détaillé.

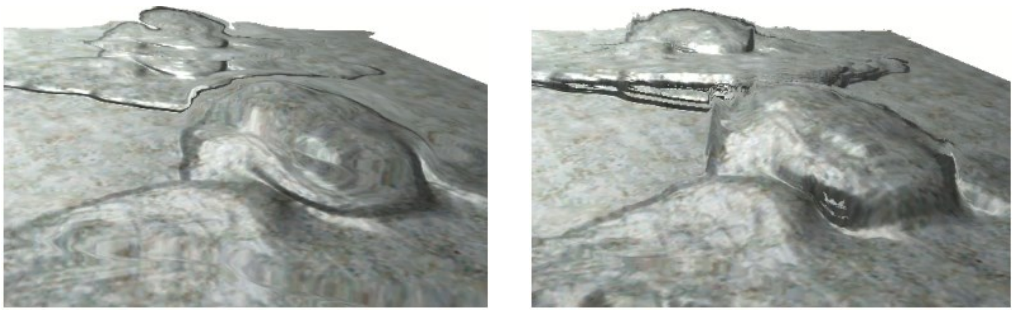
D. Placage de parallaxe itératif

Le placage de parallaxe itératif [51]–[56] est une méthode qui combine le placage de parallaxe avec d'autres techniques pour améliorer la qualité de rendu. Cette approche repose sur l'idée d'appliquer le placage de parallaxe plusieurs fois, de manière itérative, pour obtenir des résultats plus précis. Cette technique fonctionne en modifiant les coordonnées de texture

à plusieurs reprises en utilisant les informations de profondeur calculées à partir de la carte de déplacement. L'une des approches les plus simple est de répéter linéairement le placage de parallaxe simple plusieurs fois en prenant plusieurs échantillons de profondeurs [52]. Mais l'approche la plus courante est de répéter le placage de parallaxe avec pris en compte de la pente plusieurs fois [51]. À chaque itération, les coordonnées de texture sont ajustées en fonction de la profondeur et de la pente de la surface, ce qui permet de mieux simuler les variations de relief.

Après avoir réalisé le placage de parallaxe itératif dans sa forme la plus simple, une autre approche consiste à ajouter une recherche sécante (Figure 3.2d) afin de raffiner le point d'intersection du rayon de vue avec le microrelief [53]–[57] et corriger les défauts liés à l'approximation de la profondeur autour d'un point donné. L'idée de base de la recherche sécante est de partir de deux points de départ et d'interpoler linéairement entre eux pour obtenir une nouvelle estimation du point d'intersection. Cette nouvelle estimation est ensuite utilisée comme point de départ pour la prochaine interpolation linéaire, et ainsi de suite, jusqu'à ce que la position d'intersection soit approximée avec une précision suffisante. La recherche sécante est une méthode indépendante de la technique de placage de parallaxe et peut être utilisée individuellement [58]. Généralement, la recherche sécante est souvent utilisée pour résoudre les problèmes de distorsion de texture liés à l'approximation de la profondeur dans les techniques de rendu en relief. En ajustant itérativement les coordonnées de texture en fonction de la hauteur et de la pente de la surface, la recherche sécante permet d'améliorer la précision de la position d'intersection du rayon de vue avec le microrelief.

Le placage de parallaxe itératif permet de générer des rendus plus détaillés, mais son implémentation peut également être plus complexe. Il peut nécessiter des calculs supplémentaires pour ajuster les coordonnées de texture à chaque itération, ce qui peut avoir un impact sur les performances en temps réel. Cependant, cette approche peut être utilisée dans des applications où la qualité de rendu est primordiale, comme les jeux vidéo, les films d'animation, la réalité virtuelle, etc., pour obtenir des résultats visuellement convaincants.



(a) (b)

Figure 3.3 : Comparaison entre le placage de parallaxe simple (a) et le placage de parallaxe itératif (b) [51].

3.2.1.2 Placage de relief

Le placage de relief est une méthode introduite dans [59], elle est basée sur le placage de textures en relief [60]. Cette technique calcule le point d'intersection en deux étapes, dans la première étape, il détermine l'intervalle où se trouve la première intersection (Figure 3.4a), et dans la seconde étape, le point d'intersection est raffiné à l'aide d'une recherche binaire (Figure 3.4b).

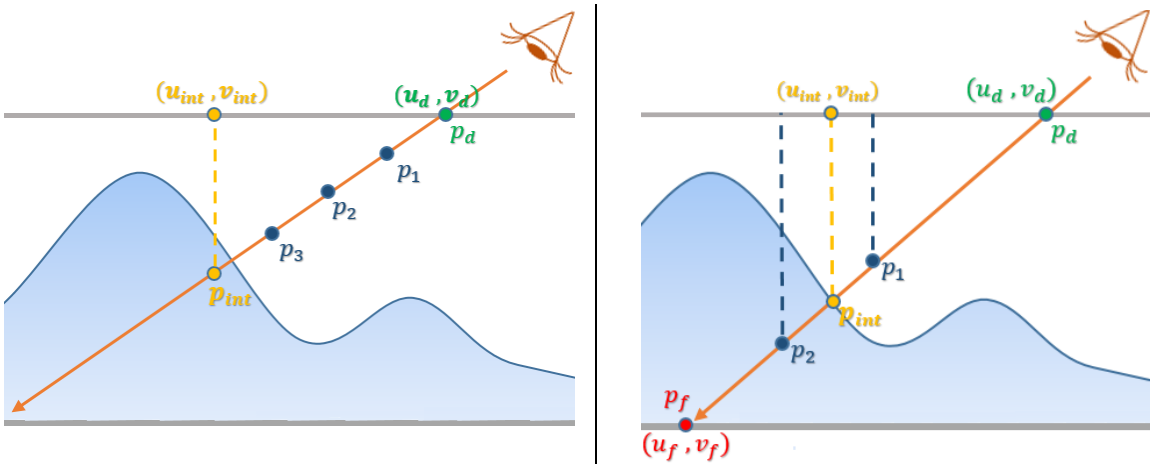


Figure 3.4 : Les deux étapes de la recherche du point d'intersection utilisées dans le placage de relief [48]. (a) Recherche linéaire. (b) Recherche binaire.

La recherche linéaire consiste à parcourir le rayon de vue à intervalles réguliers et à vérifier à chaque itération si le rayon de vue intersecte le relief stocké dans la carte de déplacement. Cette vérification peut être effectuée en comparant la profondeur du point d'intersection avec la valeur de profondeur de la carte de déplacement à cette position.

A chaque itération, la prochaine position est calculée par la relation suivante :

$$\begin{cases} (u_{i+1}, v_{i+1}, w_{i+1}) = (u_i, v_i, w_i) + pas\,.\vec{V} & Si\ d_i < D(u_i, v_i) \\ (u_{int}, v_{int}) = (u_i, v_i) & Sinon \end{cases} \quad (3.4)$$

Avec :

$$pas = 1/nb.V_z \quad (3.5)$$

Où w_i est la profondeur au point (u_i, v_i) alors que $D_i = D(u_i, v_i)$ désigne la profondeur extraite de la carte de déplacement au point (u_i, v_i). nb Représente le nombre d'itérations.

La recherche binaire consiste à subdiviser le rayon de vue en segments plus petits et à effectuer des tests d'intersection sur ces derniers. La subdivision se poursuit jusqu'à ce qu'un segment soit trouvé qui intersecte le relief ou jusqu'à ce qu'un critère d'arrêt prédéfini soit atteint. Durant la recherche, on vérifie si la position actuelle du rayon (u_i, v_i, w_i) se situe sous le relief, si c'est le cas, le rayon progresse d'un pas vers le haut, sinon il progresse vers le bas. Après chaque étape, la distance parcourue est divisée par deux pour raffiner la recherche. Le premier pas est calculé en divisant par deux la distance séparant le premier point (u_d, v_d) du point où le rayon de vue quitte la base du relief (u_f, v_f).

A chaque itération, la prochaine position est calculée par la relation suivante :

$$(u_{i+1}, v_{i+1}, w_{i+1}) = \begin{cases} (u_i, v_i, w_i) + pas_i\,.\,\vec{V} & Si\ d_i < D_i \\ (u_i, v_i, w_i) - pas_i\,.\,\vec{V} & Sinon \\ pas_{i+1} = pas_i/2 \end{cases} \quad (3.6)$$

Avec :

$$pas_0 = 1/2.V_z \quad (3.7)$$

La recherche binaire est efficace car elle réduit le nombre d'itérations nécessaires par rapport à la recherche linéaire en éliminant les segments qui ne peuvent pas percer le relief. Cela permet d'obtenir une précision élevée avec un coup relativement faible. Cependant, la

recherche binaire ne tient pas compte des profondeurs du microrelief, ce qui peut le rendre moins efficace dans le cas des microrelief irréguliers. Pour pallier à ce problème, une recherche linéaire couplée à une recherche sécante (Figure 3.2d) permet de converger encore plus rapidement [53], [55]–[57]. A l'instar de la recherche sécante, la recherche binaire une méthode indépendante et peut être utilisée individuellement pour la recherche du point d'intersection. La Figure 3.6 montre une comparaison d'un cube rendu respectivement avec la recherche sécante et la recherche binaire.



Figure 3.5 : Théière rendue avec le placage de relief en utilisant différentes cartes de déplacement [59].

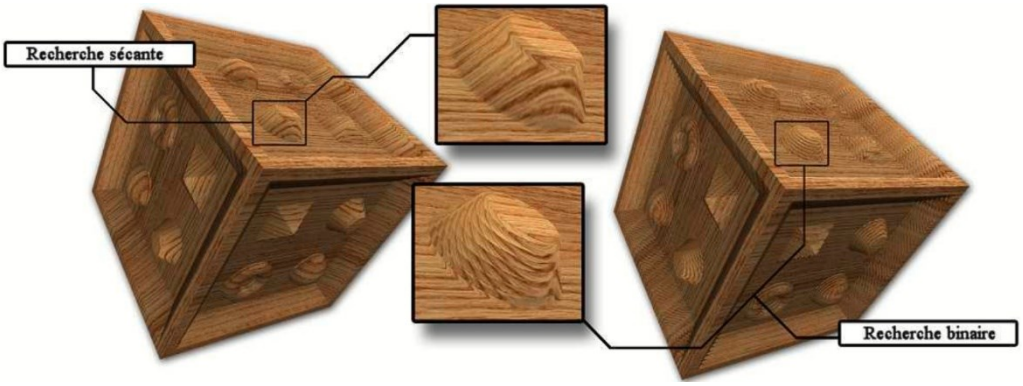


Figure 3.6 : Comparaison entre la recherche binaire et la recherche sécante [58].

3.2.2 Technique avec encodage de l'espace vide

Pour converger rapidement vers le premier point d'intersection, ces techniques reposent sur l'encodage de l'espace vide autour de chaque pixel de la carte de profondeurs dans une étape de prétraitement, et pendant la phase de rendu, cet espace est utilisé par les algorithmes de traçage de rayons pour converger plus rapidement vers le premier point d'intersection.

3.2.2.1 Traçage de sphères

Cette technique calcule l'espace autour des pixels de la carte de profondeurs sous forme de sphère [61]. C'est la première méthode qui se base sur le calcul de l'espace vide pour converger plus rapidement vers le premier point d'intersection en se basant sur le traçage de sphères [62]. Le concept est le même, mais au lieu d'appliquer l'algorithme de traçage de rayons pour les surfaces, nous l'appliquons pour le rendu à base des cartes de déplacement au niveau du GPU (Figure 3.7). Dans une phase de prétraitement, une carte de distance tridimensionnelle est calculée à partir de la carte de profondeurs pour déterminer la mesure de la distance minimale entre chaque texel par rapport au relief. Cette carte de distance stocke toutes les informations nécessaires pour converger plus rapidement vers le point d'intersection (Figure 3.7).

Durant la phase de rendu, un traçage de sphères permet à chaque itération de se rapprocher significativement de la première intersection avec le relief. La formule de recherche est la suivante :

$$(u_{i+1}, v_{i+1}, w_{i+1}) = (u_i, v_i, w_i) + R(u_i, v_i, w_i).\ \vec{V} \qquad (3.8)$$

Avec $R(u_i, v_i, w_i)$ désigne la distance ou le rayon de la sphère donnée par la carte de distance au point (u_i, v_i), et w_i est la profondeur au point (u_i, v_i).

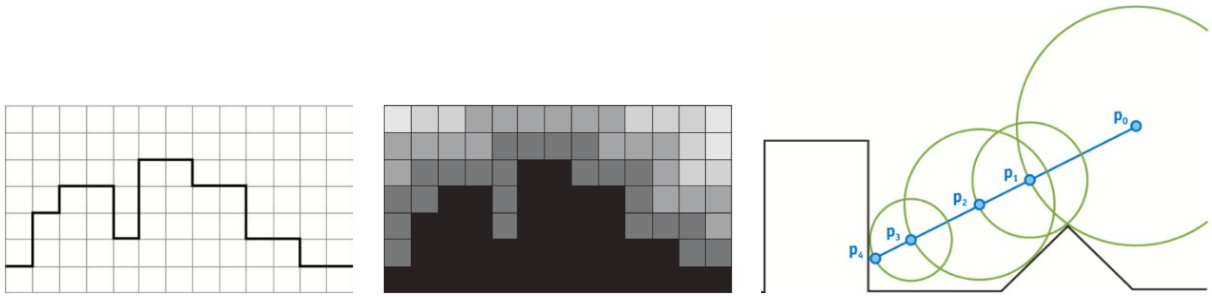


Figure 3.7 : Principe de la technique de traçage de sphères [61]. (a) La carte de profondeurs visualisée en 2D. (b) La carte de déplacement correspondante. (c) Processus de la recherche.

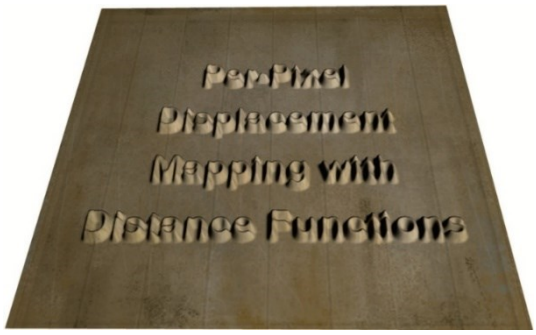


Figure 3.8 : Rendu d'un texte en 3D en utilisant la technique de traçage de sphère [61].

3.2.2.2 Traçage de cylindres

Le traçage des cylindres a été introduit dans [63]. L'étape de prétraitement de cette technique définit pour chaque pixel de la carte de profondeurs, un rayon d'un cylindre à l'intérieur duquel, aucun rayon de vue ne peut percer le relief plus d'une fois (Figure 3.9). Lors de la recherche de l'intersection, ce rayon permet d'avancer sans risquer de sauter le premier point d'intersection. Une fois ce point est trouvé, il est raffiné en utilisant une recherche binaire entre les deux dernières positions.

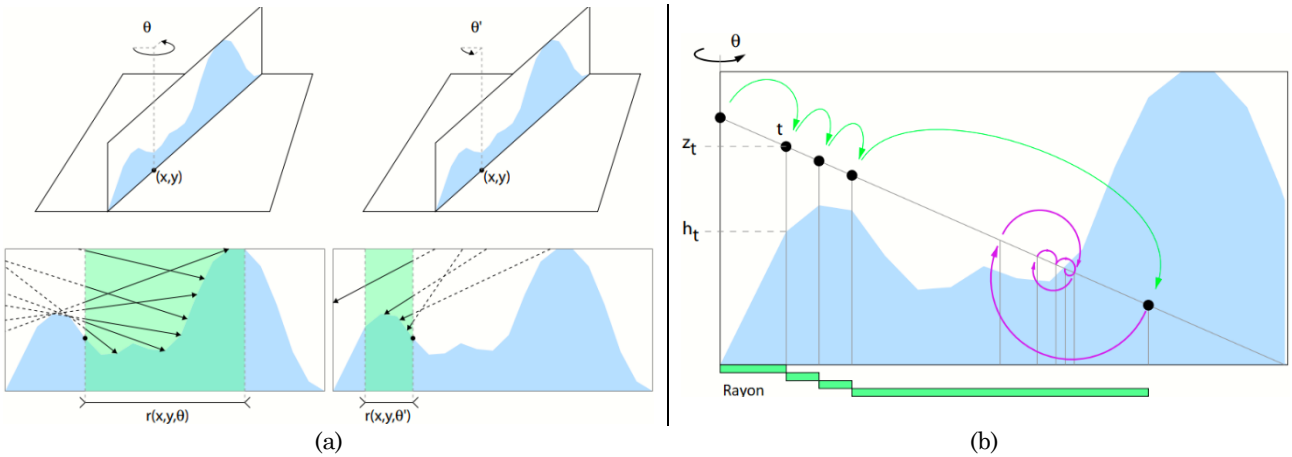


Figure 3.9 : Principe de la technique du traçage de cylindres. (a) Traçage de rayons. (b) Recherche binaire [63].

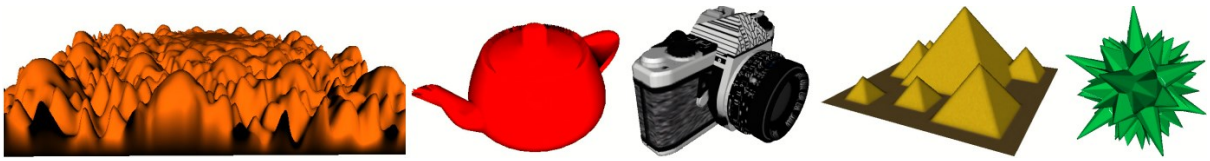


Figure 3.10 : Objets 3D rendus avec la technique du traçage de cylindres [63].

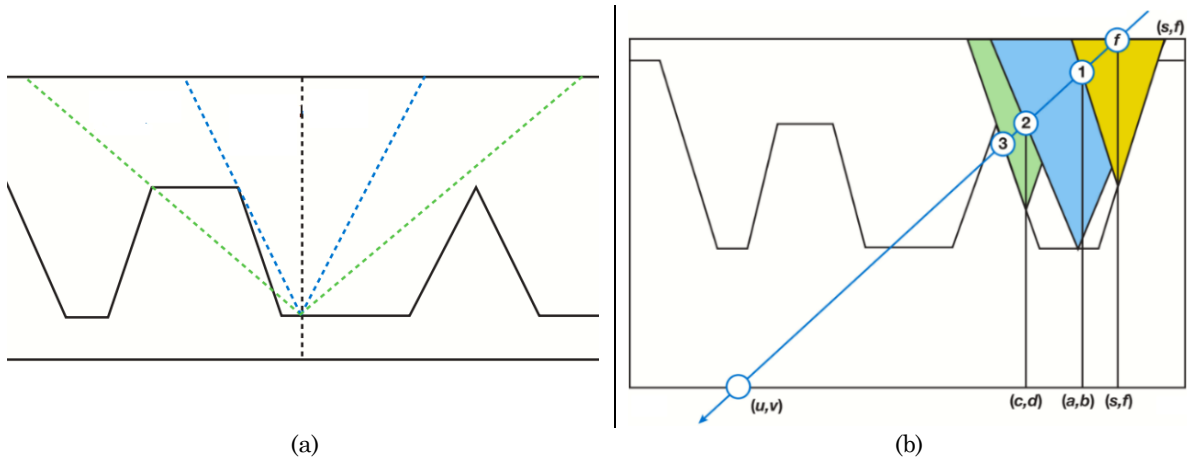
3.2.2.3 Traçage de cônes

La technique de traçage de cônes introduite dans [64]–[66] propose de calculer l'espace vide sous forme de cônes ouverts vers le haut (Figure 3.11). Les cônes sont calculés à partir de la carte de profondeurs et sont stockés dans un canal libre de la carte de déplacement en prenant leurs ratios. La technique a été proposée en deux versions, la technique conservative [64], [66] et la technique relaxée [65]. Pendant la phase de rendu, un traçage de cônes (Figure 3.11b) permet de converger plus rapidement vers le point d'intersection en utilisant la relation suivante :

$$(u_{i+1}, v_{i+1}, w_{i+1}) = (u_i, v_i, w_i) + \frac{Ratio.\, D(u_i, v_i) - w_i)}{\|\vec{V}_{xy}\| + Ratio}\vec{V} \tag{3.9}$$

Où $D(u_i, v_i)$ est la profondeur extraite de la carte de déplacement au point (u_i, v_i).

Les deux versions ont été améliorées dans [46], [67]. Ces améliorations consistent tout d'abord à utiliser des algorithmes linéaires $O(n)$ au lieu d'algorithmes quadratiques $O(n^2)$ pour calculer les cônes conservatifs et les cônes relaxés. Calculer et stocker les rayons des cônes au lieu de leurs ratios ayant ainsi des angles de cône à l'ordre de $\pi/2$ plutôt que $\pi/4$. Enfin, étendre la technique pour prendre en charge les textures non carrées en utilisant une rectification elliptique pendant la phase de rendu. La Figure 3.13 montre une comparaison de rendu de la technique de traçage de cônes.



(a) (b)

Figure 3.11 : Technique de traçage de cônes. (a) Les deux versions de cône, conservative en bleu et relaxée en vert. (b) Processus de la recherche [65].

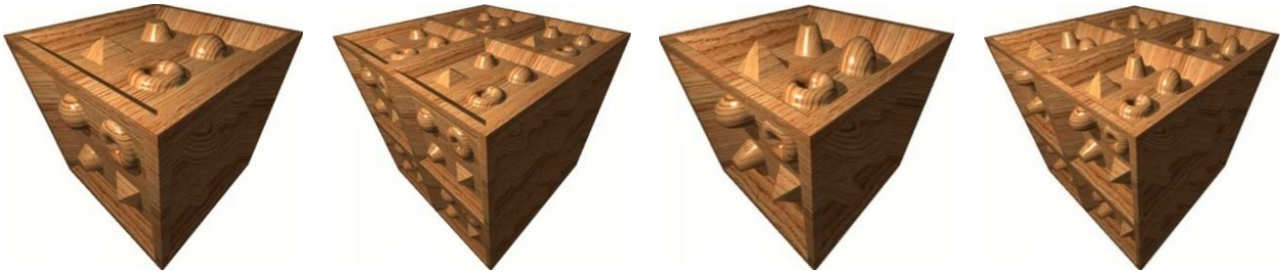
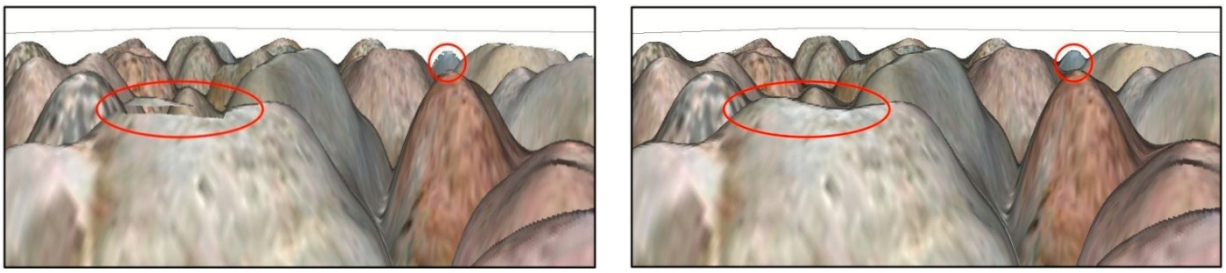


Figure 3.12 : Cube rendu avec la technique de traçage de cônes relaxés [65].



(a) (b)

Figure 3.13 : Comparaison entre deux rendus par la technique de traçage de cônes relaxés. (a) Algorithme quadratique [65]. (b) Algorithme linéaire [46].

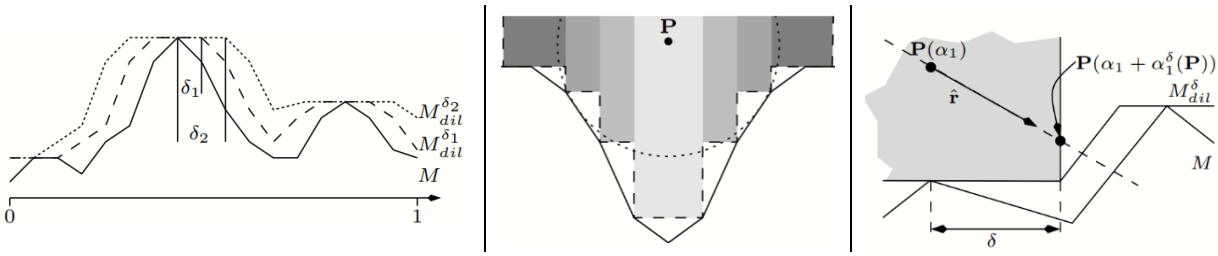
3.2.2.4 Dilatation et érosion

Une autre façon de calculer l'espace vide autour d'un texel est d'utiliser des cartes de dilatation et d'érosion [68]. Ces deux cartes sont calculées à partir de la carte de profondeurs en s'appuyant sur des opérations de filtrage du maximum et du minimum via les formules suivantes :

$$M_{dil}^{\delta} = max\{M(S') : |u-u'| < \delta \cap |v-v'| < \delta\}$$
$$M_{ero}^{\delta} = min\{M(S') : |u-u'| < \delta \cap |v-v'| < \delta\} \quad (3.10)$$

Où $S = (u, v)$ désigne la taille choisie de la région suivant les directions de u et v.

Ces deux cartes permettent d'avoir à chaque texel une région sécurisée. Les intersections successives du rayon de vision avec ces régions permettent de converger vers le point d'intersection (Figure 3.14).



(a) (b) (c)

Figure 3.14 : (a) Deux dilatations différentes avec $\delta_1 < \delta_2$. (b) Zone de sécurité pour quatre dilatations. (c) Calcul de l'intersection du rayon de vue avec la zone de sécurité [68].

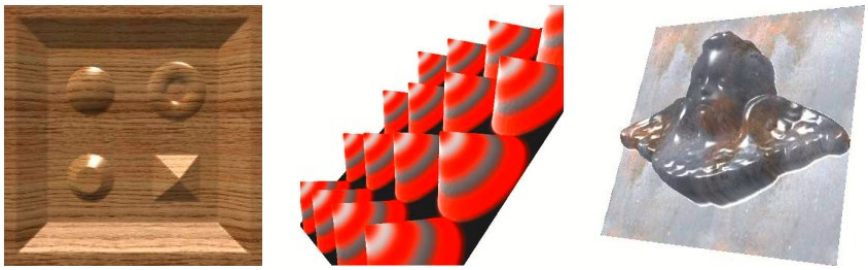


Figure 3.15 : Objets rendu à l'aide de l'encodage de l'espace vide par dilatation et érosion [68].

3.2.2.5 Placage de déplacement pyramidale

Placage de déplacement pyramidale [69]–[73] permet de créer une structure pyramidale des profondeurs en calculant à chaque fois une carte qui est quatre fois plus petite que la précédente et en prenant le maximum de la profondeur de chaque groupe de quatre pixels. Le point d'intersection entre le rayon de vue et le microrelief est obtenu par les intersections successives avec les lignes horizontales représentant la profondeur maximale de chaque

niveau de la pyramide (Figure 3.16). La carte de déplacement pyramidale est une collection d'images de résolution réduite de la carte de déplacement la plus détaillée (Figure 3.17).

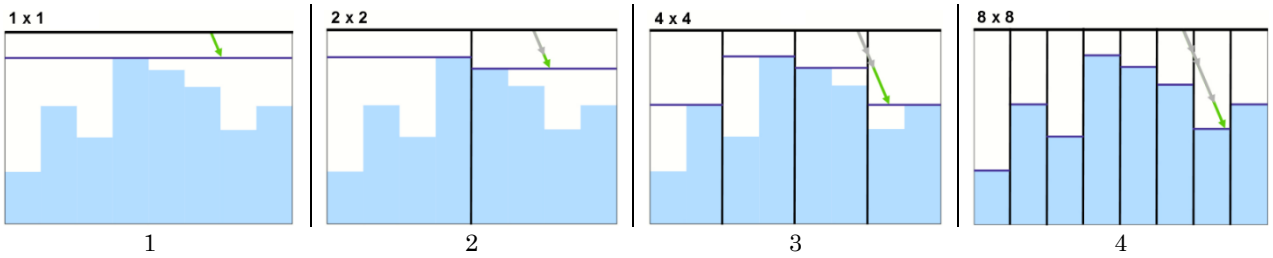


Figure 3.16 : Principe du placage de déplacement pyramidale. Un rayon est tracé hiérarchiquement dans l'ordre descendant en utilisant une structure pyramidale [70].

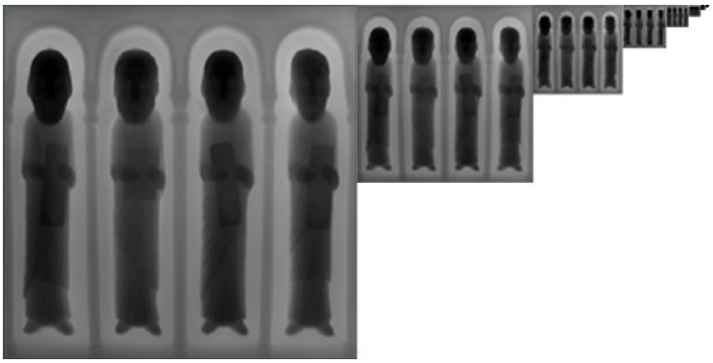


Figure 3.17 : La carte de déplacement pyramidale [69].

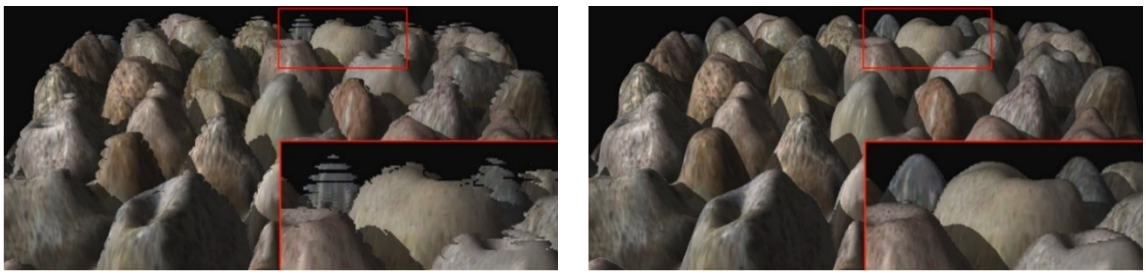


Figure 3.18 : Comparaison entre le placage de relief (a) et le placage de déplacement pyramidale (b) [70].

3.2.2.6 Placage d'extrusion et de révolution par pixel

Le placage d'extrusion par pixel [74] consiste à extruder les modèles 3D selon une forme binaire stockée dans une texture 2D sans perturber le maillage de base (Figure 3.19). L'espace vide est calculé en utilisant la Distance Transforme Euclidienne décrit dans [75]–[77] et stocké dans une texture 2D appelée carte de distance. Les normales de la forme extrudée sont calculées à partir de cette carte. La forme binaire, la carte de distance et les normales sont stockées dans une texture 2D appelée carte de forme.

Pendant la phase de rendu, un traçage de cercles horizontales dont les rayons sont récupérés de la carte de forme permet de converger plus rapidement vers le point d'intersection avec la forme extrudée. En exploitant les techniques d'extrusion et le principe du placage de déplacement par pixel, cette technique a été par la suite étendue [78] afin de créer des objets en entier et sans maillage polygonal (Figure 3.20). Des améliorations ont été proposées pour corriger le point d'intersection et pour étendre l'algorithme d'extrusion afin de créer les contours des surfaces extrudées [79], [80]. Le placage de révolution par pixel [78], [81], [82] repose sur les mêmes principes du placage d'extrusion, sauf que les objets sont créés en utilisant des algorithmes de révolution (Figure 3.21).

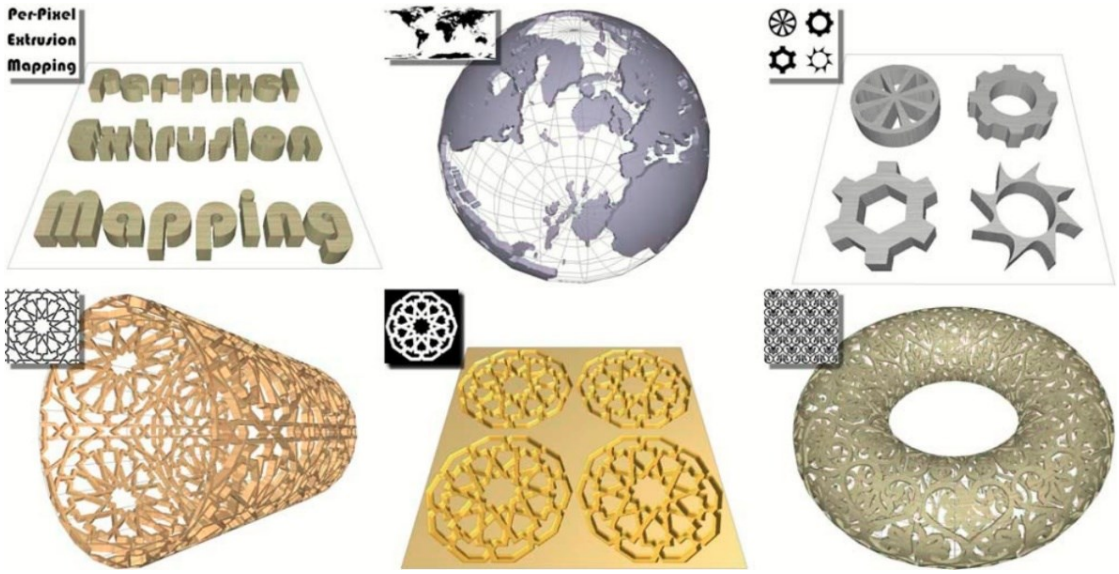


Figure 3.19 : Exemples de placage d'extrusion par pixel [74].

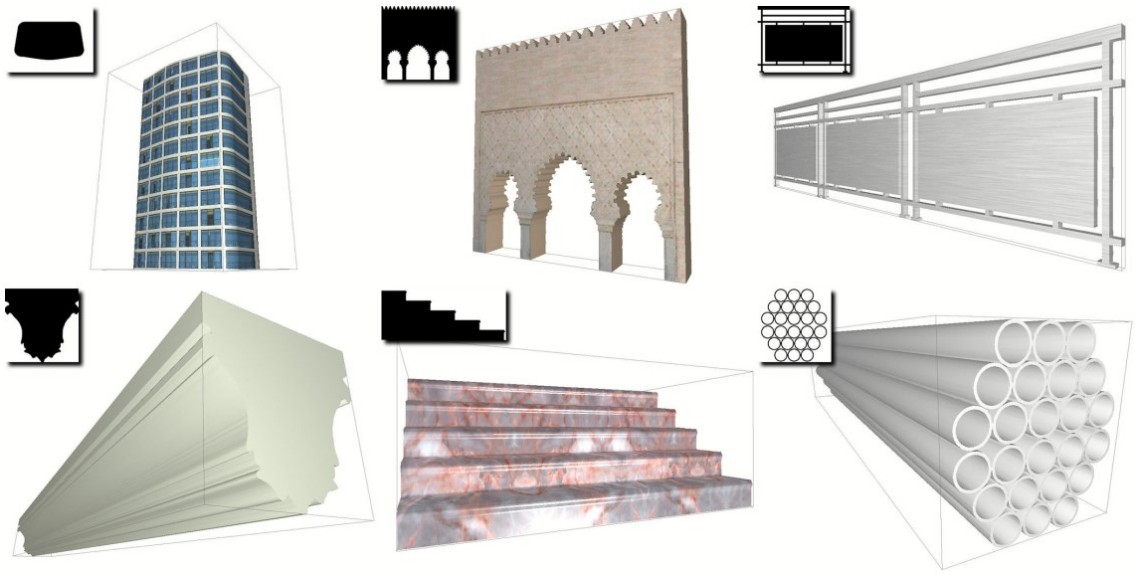


Figure 3.20 : Exemples d'objets 3D crées en entier à l'aide de l'extrusion par pixel [78].

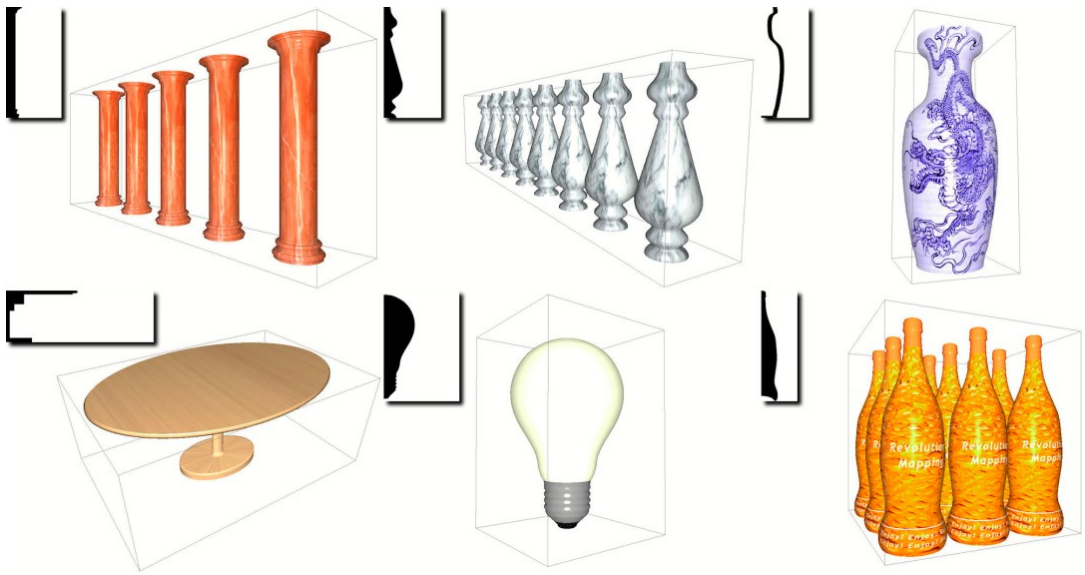


Figure 3.21 : Exemples d'objets 3D crées en entier à l'aide de la révolution par pixel [78].

3.3 TRAITEMENT DE LA SILHOUETTE

La silhouette fait référence aux bordures extérieures ou aux lignes de contour d'un objet en relief sur une surface 3D. Elle représente la limite entre la partie visible de l'objet et la partie non visible. Bien que les techniques citées dans les sections précédentes permettent d'avoir des qualités de rendu satisfaisantes, elles ne sont adaptées que pour les surfaces planes, car les algorithmes de traçage de rayons utilisés par ces techniques ne prennent pas en compte la silhouette aux bords des objets 3D.

La gestion de la silhouette nécessite des calculs supplémentaires pour prendre en compte la nature de la surface aux bords des objets 3D lors de la recherche de l'intersection du rayon de vue avec le microrelief. Pour cela, trois solutions ont été proposées. La première consiste à utiliser une représentation locale de la surface 3D à chaque sommet de la surface 3D. La deuxième effectue une extrusion du maillage polygonal, et la troisième se base sur une phase de prétraitement spécifique.

3.3.1 Représentation locale

Deux représentations locales ont été proposées. La première consiste à utiliser une approximation quadratique [83], [84]. Et la deuxième consiste à utiliser l'espace tangent [85], [86].

L'approximation quadratique est une technique utilisée dans les rendus 3D pour simplifier ou représenter une surface complexe à l'aide d'une approximation de degré deux. Elle est souvent utilisée pour obtenir des résultats en temps réel avec une complexité réduite tout en conservant la qualité visuelle de l'image finale. L'approximation quadratique présentée dans [83], [84] repose sur le calcule de deux paramètres approximatifs *a* et *b* représentant la surface 3D à chaque sommet du maillage dans une étape de prétraitement. Et pendant la phase de rendu, ces deux paramètres sont exploités par l'algorithme de traçage de rayons afin de déterminer les parties de la surface appartenant à la silhouette (Figure 3.22, Figure 3.23). Il est important de noter que l'approximation quadratique peut introduire des erreurs d'approximation et ne convient pas toujours à toutes les situations. Son utilisation appropriée dépend des spécificités de la scène et des exigences de rendu, et peut nécessiter des ajustements et des optimisations pour obtenir des meilleurs résultats.

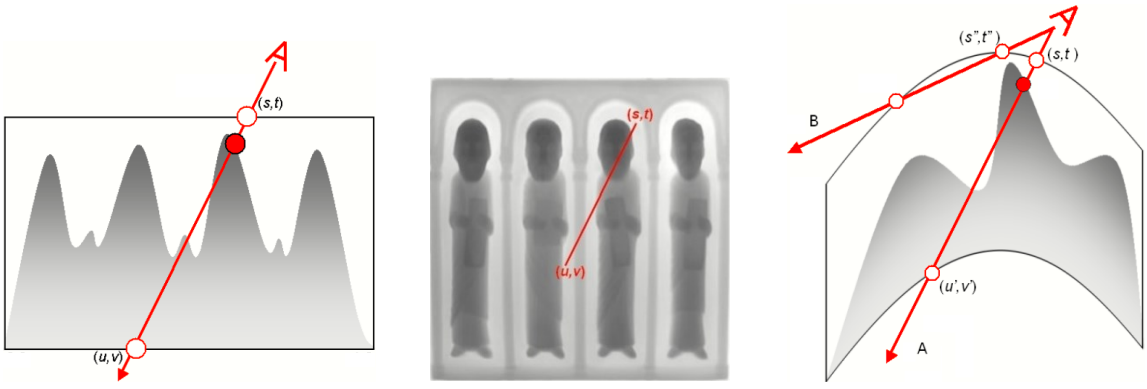


Figure 3.22 : Recherche de l 'intersection du rayon de vue avec le microrelief dans le cas d'une surface plane et dans le cas d'une surface courbée [84].

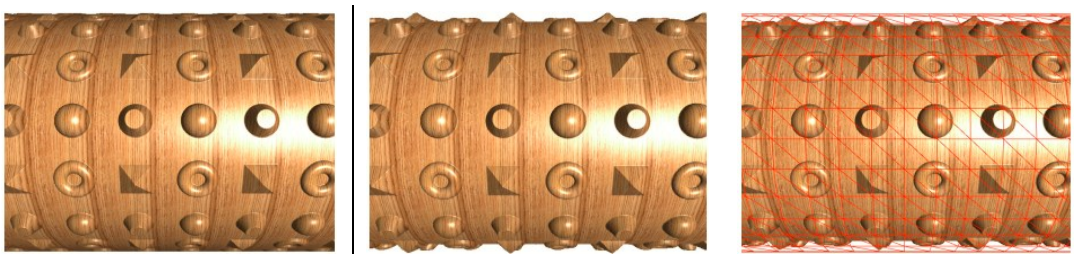


Figure 3.23 : Rendu d'un cylindre par deux techniques [84]. (a) Placage de relief. (b) Approximation quadratique.

L'espace tangent (Figure 2.19b), constitue des vecteurs normale, tangent et binormale, permet d'avoir une idée sur l'orientation de la surface à chaque sommet du maillage. L'utilisation de cet espace pour le traitement de la silhouette [85], [86] consiste à projeter les coordonnées de texture d'un point de vue sur l'espace tangent de la surface 3D afin de déterminer les nouvelles coordonnées de texture au niveau de la silhouette. Cette approximation permet de prendre en compte la courbure de la surface, et de déterminer les parties appartenant à la silhouette, contribuant ainsi à améliorer la qualité de rendu.

Le principe de base de cette approche est illustré dans la Figure 3.24, le rayon de vue est droit dans l'espace objet, mais incurvé dans l'espace texture. Si le rayon de vue est plié au fur et à mesure selon l'espace tangent, nous obtenons essentiellement une approximation linéaire par morceaux du rayon de vue incurvé dans l'espace de texture. Ainsi, si le rayon intersecte le relief stocké dans la carte de déplacement, elle est ombrée. Sinon, il appartient à la silhouette et il doit être ignoré. L'avantage d'utiliser l'espace tangent au lieu d'une approximation quadratique c'est que cet espace est toujours inclus dans le maillage 3D et ne nécessite aucun prétraitement supplémentaire.

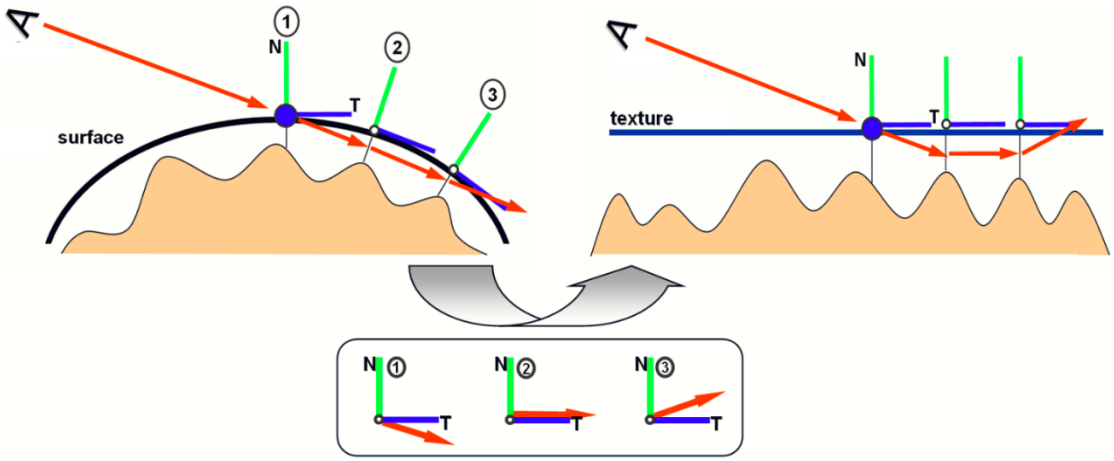


Figure 3.24 : Principe du traitement de la silhouette à base de l'espace tangent [86].



Figure 3.25 : Objets 3D rendus avec traitement de la silhouette à base de l'espace tangent [86].

3.3.2 Placage à l'aide de coquille

Le placage à l'aide de coquille procède à l'extrusion de chaque triangle du maillage selon les normales de ses trois sommets [87]. L'extrusion donne un prisme constitué de huit triangles (Figure 3.26). Pendant la phase de rendu, le traçage de rayons est effectué à l'intérieur de ce prisme (Figure 3.26c), et dans le cas où le rayon quitte le prisme avant d'intersecter le relief, alors le pixel traité fait partie de la silhouette. Pour éviter certains défauts de discontinuité liés à l'interpolation bilinéaire, le prisme est subdivisé en trois tétraèdres à l'aide d'un algorithme décrit dans [88]. L'utilisation des coordonnées barycentriques introduites dans [89] permet de définir une relation entre chaque point 3D contenu dans le prisme, et un seul texel dans la carte de déplacement. L'utilisation de textures 3D semi-transparentes permet de supporter certaines fonctionnalités plus avancées [90]. Une fonction de lissage couplée aux patchs de cônes permet d'éliminer fortement les distorsions, et produit ainsi des résultats très satisfaisants [91]. L'un des principaux inconvénients de l'utilisation de la coquille est l'augmentation de la densité du maillage, qui peut être multipliée par huit ou plus, ce qui limite son utilisation pour le rendu en temps réel.

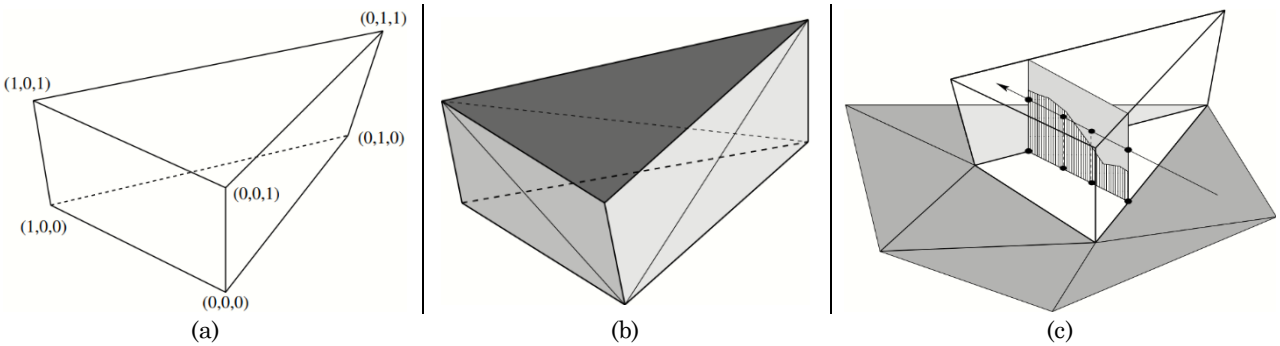


Figure 3.26 : Principe du placage à l'aide de coquille [87]. (a) Vecteurs utilisés pour définir le deuxième système de coordonnées. (b) Prisme avec ses triangles résultants. (c) Traçage de rayons dans le prisme extrudé.

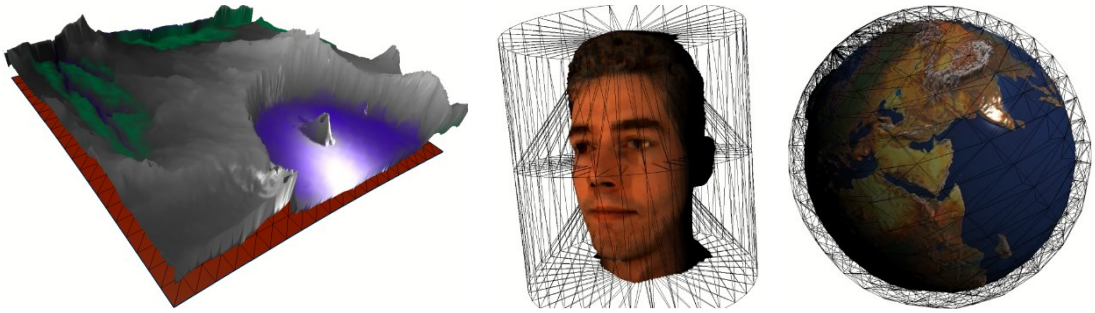


Figure 3.27 : Objets rendus avec le placage à l'aide de coquille [87].

3.3.3 Placage de déplacement dépendant du point de vue

Le placage de déplacement dépendant du point de vue [92] consiste à calculer, pour chaque rayon de vue, la distance entre chaque point d'un polygone et la surface de déplacement (Figure 3.28). Pour pouvoir gérer la silhouette, la courbure de la surface de base doit également être pris en considération. Une fonction à cinq dimensions appelée VDM est ainsi définie pour stocker les coordonnées de texture, les coordonnées sphériques du rayon de vue, et l'indice de courbure de la surface le long du rayon de vue. Cette fonction représente une grande quantité de données, pour cette raison, elle est compressée et stockée sous forme

d'une texture 3D (Figure 3.29). Cette technique a été généralisée afin de pouvoir gérer les cartes de profondeurs 3D, et pour limiter certaines distorsions [93].

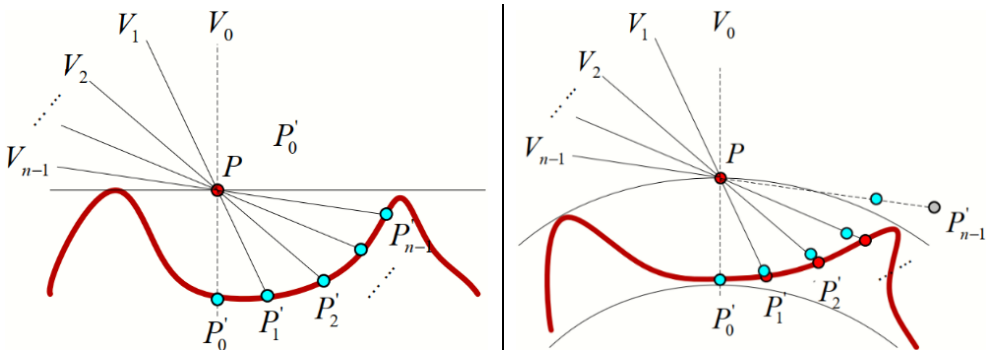


Figure 3.28 : Variation de déplacement en fonction de la vue et la nature de la surface [92]. (a) Surface plane. (b) Surface courbée.

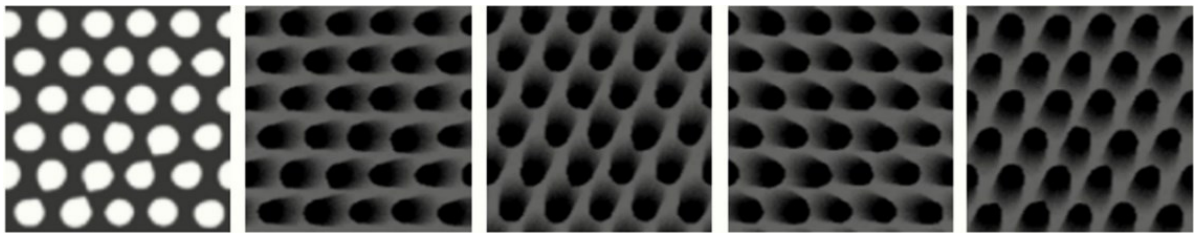


Figure 3.29 : Images VDM synthétisées à partir d'une carte de profondeurs [92]. (a) La carte de profondeurs. (b-e) Images VDM pour différentes directions de vue.



Figure 3.30 : Objets rendus avec le placage de déplacement dépendant du point de vue [92].

3.4 AUTO-OMBRAGE

L'auto-ombrage est une technique de rendu utilisée pour simuler les ombres générées par un objet sur lui-même en fonction de la position de la source de lumière dans la scène. Cela permet de rendre les objets plus réalistes en prenant en compte la manière dont ils interagissent avec la lumière et projettent des ombres sur leurs propres surfaces. Sans auto-ombrage, les objets paraîtraient plats et sans vie, car il n'y aurait pas de variations d'ombres sur leur propre surface. Il est donc un élément clé pour créer des images 3D crédibles.

Pour simuler l'auto-ombrage, on utilise différentes techniques de rendu, notamment les techniques à base de traçage de rayons. Ces techniques permettent de simuler la propagation de la lumière dans une scène 3D de manière très précise, en prenant en compte les interactions entre la lumière, les objets et les surfaces. Il existe différentes techniques pour simuler l'auto-ombrage, chacune avec ses avantages et ses inconvénients en termes de coût de calcul et de précision.

3.4.1 Placage des ombres

Le placage des ombres permet de simuler les ombres portées créées par les objets dans une scène en utilisant une carte d'ombres. Cette technique a été introduite dans [94] et est utilisée par des techniques du placage de déplacement par pixel [59], [92], [95] afin de simuler les ombres générés par les reliefs sur l'objet lui-même. Le processus commence par le calcul d'une carte d'ombres pour chaque source de lumière dans la scène. Ce calcul repose sur le traçage d'un rayon depuis la source de lumière vers la surface de l'objet la plus proche, et si ce rayon intersecte un objet sur son chemin, on considère que le pixel correspondant est dans l'ombre et on le marque comme tel dans la carte d'ombres. Lors du rendu de la scène, on utilise cette carte pour déterminer si un pixel est dans l'ombre ou éclairé par la source de lumière (Figure 3.31b).

Le placage des ombres est une technique relativement simple et efficace pour simuler les ombres portées dans une scène 3D, mais elle peut produire des artefacts visuels si la résolution de la carte d'ombres est trop faible ou si les objets sont très proches les uns des autres. Pour éviter ces problèmes, ils existent plusieurs variantes et améliorations comme l'utilisation de techniques à base de filtre pour adoucir les transitions entre les zones d'ombres et d'éclairage, ou l'utilisation de plusieurs cartes d'ombres pour améliorer la résolution et la qualité des ombres.



(a) (b)

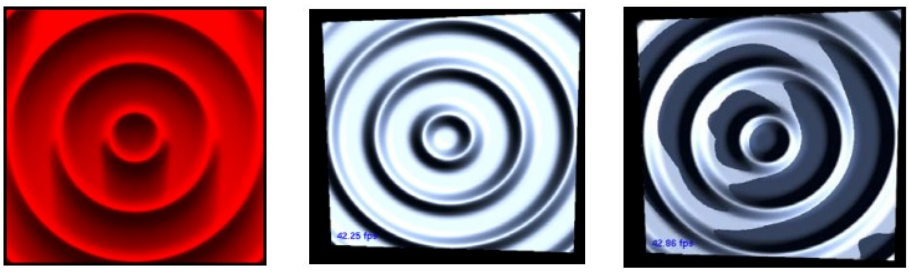
Figure 3.31 : Auto-ombrage avec la technique du placage des ombres [59]. (a) Traçage de rayons de lumière. (b) Placage de relief avec pris en charge de l'auto-ombrage.

3.4.2 Placage d'horizon

Le placage d'horizon [96], [97] permet de générer des ombres en utilisant une carte d'horizon précalculée. Cette carte représente l'ensemble des directions de visibilité depuis chaque point de l'environnement 3D, c'est-à-dire les directions dans lesquelles il n'y a aucun obstacle visible depuis ce point. Lors du rendu, la carte d'horizon est utilisée pour déterminer les ombres portées sur chaque pixel de l'image. Pour chaque pixel, la direction de visibilité est

calculée en utilisant les normales de la surface et la position de la caméra. Ensuite, la carte d'horizon est interrogée pour savoir si cette direction est bloquée par un objet, si c'est le cas, le pixel est ombré, sinon il est éclairé par la source lumineuse (Figure 3.32).

Cette technique présente l'avantage d'être relativement simple à mettre en œuvre et de déterminer les ombres avec peu de calculs. Cependant, elle nécessite une étape de prétraitement coûteuse pour générer la carte d'horizon, ce qui peut limiter son utilisation dans des environnements dynamiques ou pour des applications en temps réel.

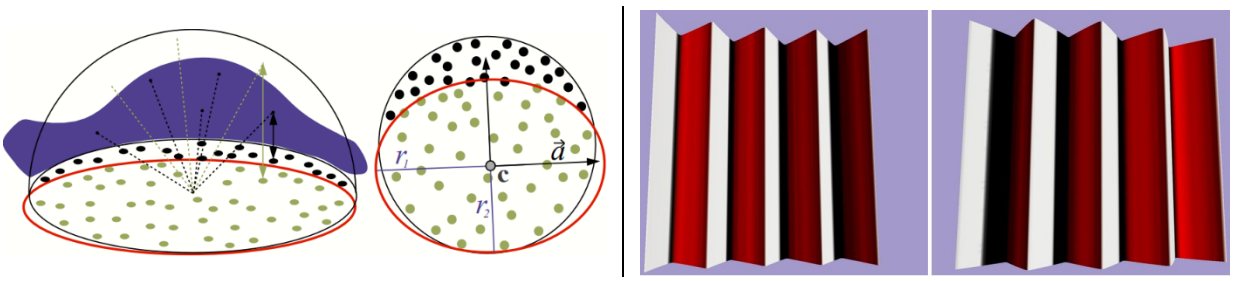


(a) (b) (c)

Figure 3.32 : (a) Carte d'horizon. (b) Placage de bosselures. (c) Placage d'horizon [96], [97].

3.4.3 Auto-ombrage par carte de visibilité

L'auto-ombrage par carte de visibilité [98] est une technique qui permet de générer des ombres en utilisant une carte de visibilité précalculée. Cette carte représente l'ensemble des points visibles, c'est-à-dire les points qui ne sont pas cachés par un objet depuis ce point. Le processus commence par le calcul des zones d'ombre et de lumière en fonction de la position de la source de lumière, de la caméra et des objets dans la scène 3D (Figure 3.33a). Ensuite, des cartes de visibilité sont créées pour chaque objet enregistrant l'étendue de l'ombre qu'il projette sur d'autres objets dans la scène. Lors du rendu, la carte de visibilité est interrogée pour savoir si un pixel dans la scène est bloqué par un objet, si tel est le cas, le pixel est ombré, sinon il est éclairé par la source lumineuse (Figure 3.33b).



(a) (b)

Figure 3.33 : (a) Auto-ombrage par carte de visibilité [98]. (b) Rendu basé sur une carte de visibilité.

3.4.4 Fonction d'illumination globale

La fonction d'illumination globale est une technique avancée qui permet de simuler de manière réaliste l'interaction de la lumière avec les différents objets d'une scène 3D et peut être utilisée par les techniques de placage de déplacement par pixel pour la simulation de l'auto-ombrage. Contrairement aux techniques de rendu simples qui se contentent d'ajouter une source de lumière statique, l'illumination globale prend en compte les interactions

complexes entre la lumière et les différentes natures de relief au niveau des surfaces 3D. Elle est capable de simuler des effets de réflexion et de réfraction très réalistes sur les microreliefs d'un objet 3D, mais elle peut être très gourmande en temps de calcul.

Comme il été mentionné dans le chapitre précédent, l'illumination globale peut être réalisée à l'aide de différents algorithmes de calcul, tels que le traçage de rayons, le traçage de chemins, le placage de photon, la radiosité, etc. Ces différents algorithmes offrent des avantages et des inconvénients en termes de temps de calcul et de qualité d'image, et leur choix dépend des besoins spécifiques de la scène.

3.5 CONCLUSION

Le placage de déplacement par pixel est une technique qui permet de simuler les effets de relief et de détails sur les surfaces en utilisant des algorithmes de traçage de rayons et des textures spéciales appelées cartes de déplacement. Contrairement aux textures de couleur, les cartes de déplacement stockent respectivement les normales et les profondeurs représentants les informations de relief pour chaque pixel de l'image.

Afin de créer un objets 3D par la technique de déplacement par pixel, il faut prendre en considération la simulation des effets de relief, le traitement de la silhouette et la pris en charge de l'auto-ombrage. La simulation des effets de relief est réalisée par l'exécution des algorithmes de traçage de rayons afin de trouver le point d'intersection entre le rayon de vue et le microrelief stocké dans la carte de déplacement. Le traitement de la silhouette consiste à prendre les surfaces courbées en considération par les algorithmes de traçage de rayons afin de déterminer les parties appartenant à la silhouette aux bords de l'objet 3D. Et l'auto-ombrage consiste à simuler les ombres générées par un objet sur lui-même en prenant en compte les différentes sources de lumière dans la scène.

Le placage de déplacement par pixels est une technique qui offre une grande précision et une grande flexibilité, permettant de simuler une grande variété de surfaces. Cependant, il peut être coûteux en termes de ressources de traitement, en particulier pour les scènes qui sont très complexes. De plus, cette technique peut nécessiter des maillages spécialisés et un pipeline de rendu optimisé pour obtenir des résultats optimaux. Malgré ça, elle est souvent utilisée dans une large gamme d'applications 3D pour ajouter des détails géométriques et améliorer la qualité visuelle du rendu en temps réel. Elle offre des avantages en termes de performance, de flexibilité artistique, de précision des détails et de compatibilité avec d'autres techniques de rendu par rapport au placage de déplacement par sommet. Cependant, il convient de noter que chaque technique a ses propres avantages et ses propres limitations, et le choix entre le placage de déplacement par pixels et le placage de déplacement par sommet dépendra des besoins spécifiques d'un projet donné.

Chapitre 4

PLACAGE DYNAMIQUE DE DEPLACEMENT PAR PIXEL AVEC TRAITEMENT DE LA SILHOUETTE

4.1 INTRODUCTION

Le placage de déplacement par pixel [45] est une technique de placage de texture qui ajoute l'effet de microrelief aux surfaces 3D sans augmenter la densité de leurs maillages correspondants. Cette technique repose sur des algorithmes de traçage de rayons pour trouver le point d'intersection entre le rayon de vue et le microrelief stocké dans une texture 2D appelée carte de déplacement. Comme il a été mentionné dans les chapitres *Chapitre 2* et *Chapitre 3* et, le principal problème du placage de déplacement par pixel est de trouver le premier point d'intersection entre le rayon de vue et le microrelief stocké dans la carte de déplacement. Comme le montrent la Figure 4.1, la première intersection est représentée par le point P le long du rayon de vue $\vec{V}$ selon la relation suivante :

$$P = Vt \tag{4.1}$$

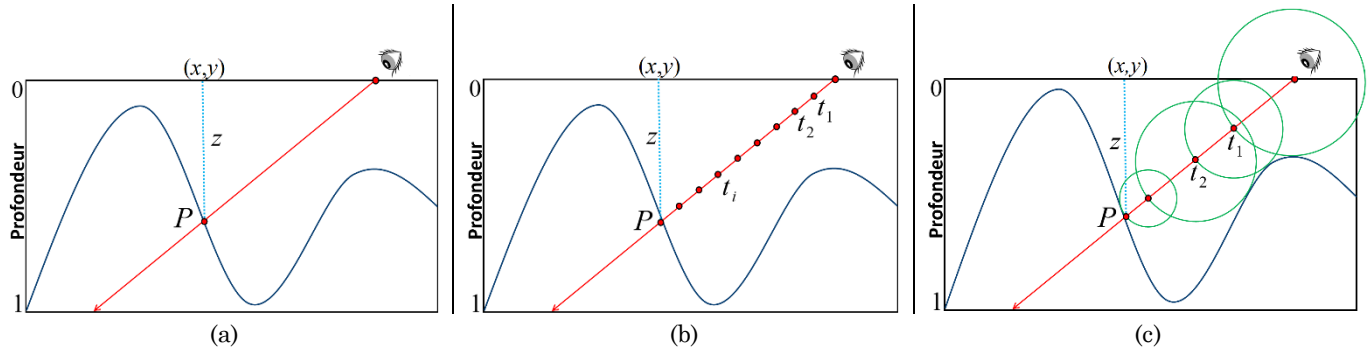


Figure 4.1 : Recherche du point d'intersection du rayon de vue avec le microrelief. (a) Première intersection. (b) Recherche en utilisant des pas fixes. (c) Recherche en utilisant des pas variables.

Les meilleures techniques du placage de déplacement par pixel utilisent des algorithmes itératifs. Pour mieux localiser la première intersection, le nombre d'itérations est prédéfini à l'avance. La taille du pas de déplacement est définie en fonction de la technique utilisée, qui peut être constante (sans encodage de l'espace vide, Figure 4.1b) ou variable (avec encodage de l'espace vide Figure 4.1c). Lors de la recherche de l'intersection, on utilise les formules générales suivantes :

$$P_{i+1} = vt_{i+1} \tag{4.2}$$

$$t_{i+1} = t_i + pas \tag{4.3}$$

Où v est le vecteur de rayon de vue V normalisé et exprimé dans l'espace de texture ayant une profondeur normalisée (c'est-à-dire v/v_z) impliquant $v_z = 1$.

Ces deux formules sont utilisées ensemble pour déterminer le point P_{i+1}. A l'itération $i+1$, le paramètre t_{i+1} permet de déterminer la position du point P avec le vecteur v, et le paramètre pas permet de déterminer son déplacement dont la valeur est calculée selon la technique utilisée. Parmi les techniques les plus répondues, en trouve la technique de placage de relief et la technique de traçage de cônes.

La technique de placage de relief est une technique très simple qui repose sur la détermination préalable du nombre d'itérations. Dans le cas des microreliefs de faibles profondeurs, un nombre minimal d'itérations est largement suffisant pour avoir une qualité de rendu satisfaisante. Mais dans le cas des reliefs de grandes profondeurs, la technique a

besoin d'un nombre d'itérations très grand que ce soit pour la recherche linéaire ou la recherche binaire. Alors il faut trouver un moyen qui permet de déterminer le nombre d'itérations selon l'échelle de profondeur des microreliefs. La technique de traçage de cônes quant à elle, repose sur l'encodage de l'espace vide autour de chaque pixel de la carte de profondeurs. Lors de l'étape de prétraitement, cet espace est encodé sous forme de cônes ouverts vers le haut puis stockés dans la carte de déplacement, et lors de l'étape de rendu, il est utilisé pour converger plus rapidement vers le point d'intersection. La technique de traçage de cônes produit des résultats satisfaisants dans le cas des surfaces planes, mais lorsqu'il s'agit des surfaces courbées, elle ne gère pas la silhouette aux bords du maillage 3D, c'est-à-dire que les reliefs se confondent avec la surface de l'objet, et dans ce cas, il ne sera pas rendu correctement. Pour surmonter ces limitations, nous allons présenter dans ce chapitre les contributions suivantes :

- Une nouvelle méthode de placage de relief qui repose sur la synchronisation entre l'échelle de profondeur, l'ongle de vue et le nombre d'itérations. Cela permettra d'optimiser le n'ombre d'itérations selon la nature du microrelief.
- Proposition d'une troisième technique de traçage de cônes qui permet d'approcher les résultats des deux techniques conservative et relaxée.
- Deux nouveaux algorithmes de traçage de cônes qui permettent de prendre en considération la courbure de la surface 3D pour déterminer les fragments appartenant à la silhouette. Ces deux algorithmes sont basés sur une approximation quadratique de la géométrie de l'objet à chaque sommet du maillage 3D.

Dans ce chapitre, toutes les expérimentations et toutes les figures sont réalisées en utilisant la configuration matériel *Config.1* (Tableau 0.2). La plupart des algorithmes sont présentés en pseudo-code, et leurs organigrammes respectifs sont présentés dans *Annex B*.

4.2 PLACAGE DYNAMIQUE DE RELIEF

Pour que la technique de placage de relief soit optimale pour le rendu en temps réel, le nombre d'itération doit être choisi d'une manier dynamique afin de prendre en compte le changement de l'échelle de profondeur et l'ongle de vue. Dans cette section nous allons présenter une nouvelle technique du placage de relief qui permet une synchronisation combinée entre l'échelle de profondeur, l'ongle de vue et le nombre d'itération.

4.2.1 Placage de relief

Le placage de relief introduit dans [59] consiste à afficher des détails très fins ou des microreliefs présents dans un grand nombre de surfaces sans augmenter la densité de leurs maillages polygonaux associés. La technique commence par une recherche linéaire (Figure 3.4a) pour localiser l'intervalle où se trouve la première intersection. Ensuite, elle utilise une recherche binaire (Figure 3.4b) en divisant successivement cet intervalle par deux, afin de mieux localiser la meilleure intersection.

La recherche linéaire commence depuis le pixel courant (u, v), et avance avec des pas réguliers suivant la direction de vue V. Cette opération est répétée tant que la position actuelle est au-dessus du relief. L'idée de la recherche binaire est de tester si la position actuelle P_i se trouve sous le relief. Dans ce cas, la progression se fait d'un pas vers le haut. Sinon, la progression sera effectuée vers le bas. Le pas est divisé par deux après chaque étape. Cette technique dont le rendu se calcule dans l'espace texture est adaptée pour le rendu temps réel, ce qui permet d'augmenter le réalisme de ce dernier en utilisant des surfaces plus complexes et plus riches en détails. Cependant, lorsque l'échelle de profondeur est assez grande, ou lorsque l'axe de vue rase la surface, les défauts d'artefacts deviennent nettement visibles comme montre la Figure 4.2.

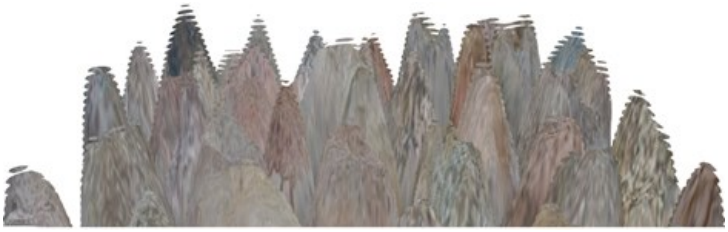


Figure 4.2 : Placage de relief avec une échelle de profondeur égale à 0.8 et un ongle de vue rasant la surface.

4.2.2 Traçage dynamique de rayon

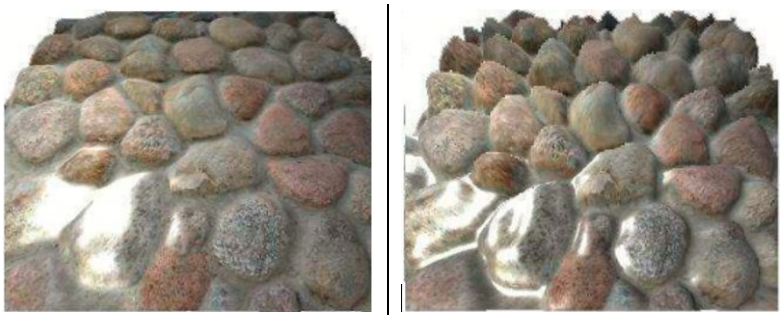
4.2.2.1 Synchronisation de l'ombrage avec l'échelle de profondeur

Cette synchronisation a été introduite dans [99], elle consiste à synchroniser l'ombrage avec la modification en temps réel de l'échelle de profondeur en mettant à jour la normal du relief du pixel courant via la relation suivante :

$$\vec{N}_b(i,j) = \frac{\left(\frac{b}{a}N_{a_x}, \frac{b}{a}N_{a_y}, \frac{b}{a}N_{a_z}\right)}{\left\|\left(\frac{b}{a}N_{a_x}, \frac{b}{a}N_{a_y}, \frac{b}{a}N_{a_z}\right)\right\|} \tag{4.4}$$

Avec a et b sont respectivement l'ancien et le nouveau facteur de l'échelle de profondeur. $\overrightarrow{N_a}$ et $\overrightarrow{N_b}$ sont respectivement l'ancien et la nouvelle normale.

L'expression (4.4) démontre qu'il suffit de multiplier les composantes x et y de la normale récupérée à partir de la carte de déplacement par la constante b/a, puis de renormaliser le vecteur ainsi obtenu. La Figure 4.3 montre un exemple de rendu sans et avec synchronisation.



Sans synchronisation Avec synchronisation

Figure 4.3 : Placage de relief avec une synchronisation entre l'échelle de profondeur et l'ombrage [99].

4.2.2.2 Synchronisation du nombre d'itérations avec l'ongle de vue

Cette synchronisation a été présentée dans [53], elle consiste à synchroniser le nombre d'itérations nb pendant la recherche de l'intersection avec le changement de la direction de vue. Pour faire cela, il faut déterminer l'ongle entre le rayon de vue $\vec{V}$ et la normal $\vec{N}$ au pixel courant, cela revient à utiliser le produit scalaire entre $\vec{V}$ et $\vec{N}$. La relation de synchronisation est la suivante :

$$nb = nb_{min} + \vec{V}.\vec{N}(nb_{min} - nb_{max}) \tag{4.5}$$

Il s'agit d'une relation d'interpolation entre le nombre minimal nb_{min} et le nombre maximale nb_{max} . La Figure 4.4 montre un ensemble d'images rendues sans et avec synchronisation.



Figure 4.4 : Placage de relief à une échelle de profondeur égale à 0.6 et avec deux ongles de vue différents, sans synchronisation (en haut) et avec synchronisation (en bas).

4.2.2.3 Synchronisation du nombre d'itérations avec l'échelle de profondeur

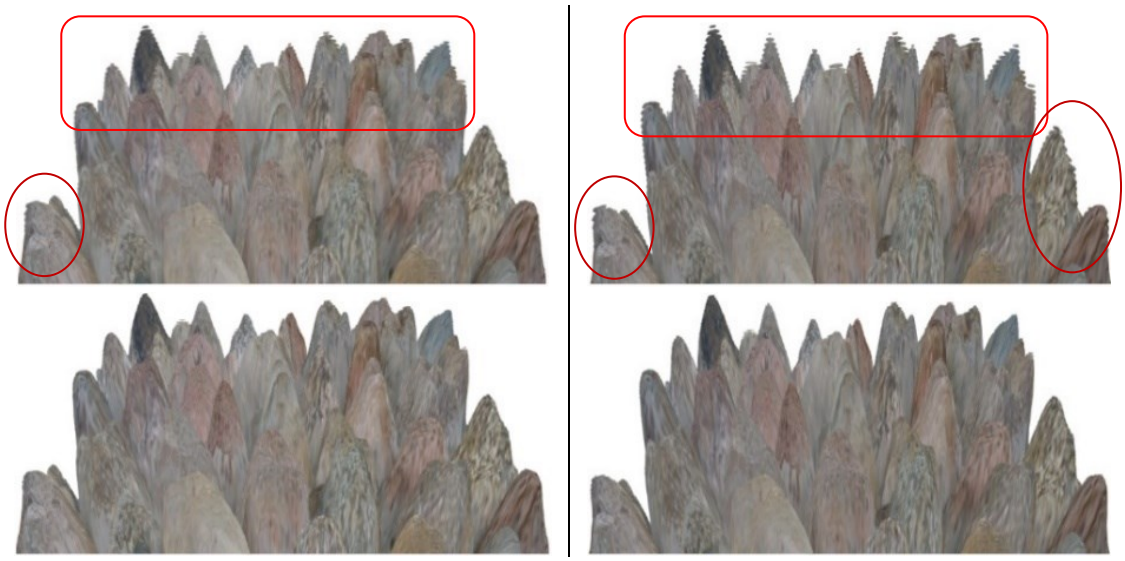
Cette synchronisation consiste à synchroniser le nombre d'itération nb pendant la recherche de l'intersection avec le changement de l'échelle de profondeur. Alors nous proposons d'utiliser la même relation (4.5) en remplaçant le produit scalaire $\vec{V}.\vec{N}$ par le facteur d'échelle a :

$$nb = nb_{min} + a(nb_{min} - nb_{max}) \tag{4.6}$$

La Figure 4.5 montre un ensemble d'images rendues sans et avec synchronisation.

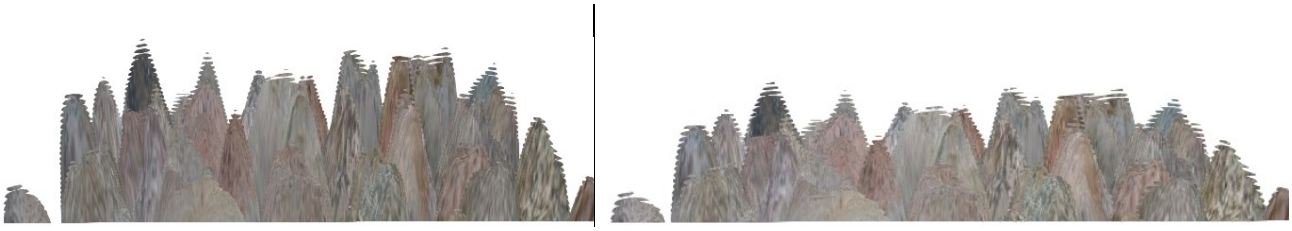
4.2.2.4 Synchronisation combinée entre le facteur d'échelle, l'ongle de vue et le nombre d'itérations

Dans la première synchronisation avec l'ongle de vue, les images sont prises avec une échelle de profondeur fixe, alors si l'échelle de profondeur augmente, nous risquons d'avoir des défauts (Figure 4.6a). La même chose dans la synchronisation avec l'échelle de profondeur, les images sont prises avec un ongle de vue fixe, alors si la direction de vue rase la surface, les défauts deviennent nettement visibles (Figure 4.6b).



0.8 1.0

Figure 4.5 : Placage de relief avec un ongle de vue fixe et deux échelles de profondeur (0.8 et 1.0), sans synchronisation (en haut) et avec synchronisation (en bas).



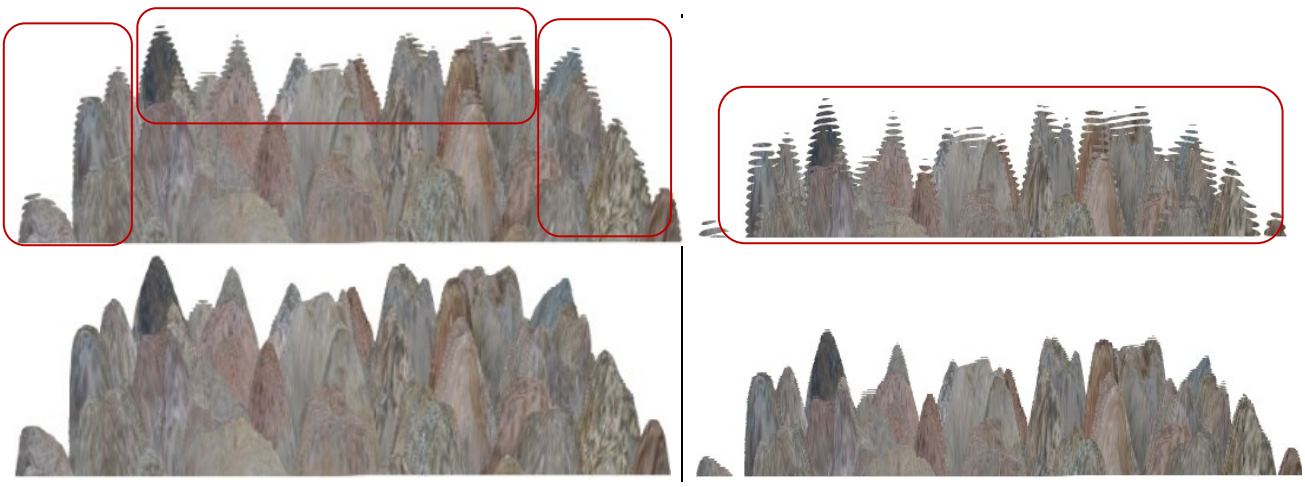
(a) (b)

Figure 4.6 : (a) Synchronisation avec l'ongle de vue à une échelle de profondeur égale à 1.0. (b) Synchronisation avec l'échelle de profondeur avec une direction de vue rasant la surface.

Pour résoudre ce problème, nous proposons de combiner les deux synchronisations dans une seule en utilisant la même relation d'interpolation linéaire :

$$nb = nb_{min} + min(V.N + a, 1.0).(nb_{min} - nb_{max}) \quad (4.7)$$

Cette relation permet d'avoir une bonne interpolation qui prend en compte l'échelle de profondeur et l'ongle de vue. Alors dans le cas où l'échelle de profondeur est assez grande et la direction de vue rase la surface, l'interpolation donne un nombre d'itération optimale. La Figure 4.7 montre un des exemples de rendu avec cette approche.



0.8 1.0

Figure 4.7 : Placage dynamique de relief en utilisant deux échelle de profondeur (0.8 et 1.0) et deux angles de vue. Sans synchronisation (en haut) et avec synchronisation (en bas).

4.3 TECHNIQUE DE TRAÇAGE DE CONES HYBRIDES

Le traçage de cônes est l'une des meilleures techniques pour le rendu de microreliefs en temps réel. En plus de la technique conservative et la technique relaxée, nous allons présenter dans cette section une troisième appelée la technique hybride.

4.3.1 Les techniques conservatives et relaxées

La technique du traçage de cônes [46], [67] consiste à calculer l'espace vide autour de chaque pixel de la carte de profondeurs dans une étape de prétraitement sous forme de cônes dont les rayons sont stockés dans le canal alpha de la carte de déplacement. Ensuit cet espace est utilisé pendant la phase de la recherche de l'intersection pour converger plus rapidement vers le point d'intersection du rayon de vue avec le microrelief. Cette technique a été présentée en deux versions, la technique conservative et la technique relaxée.

Dans la technique conservative, le cône est défini de tel sorte que le rayon de vue ne perce pas le relief (Figure 4.8a), et dans la technique relaxée (Figure 4.8b), le cône est défini de tel sorte que le rayon de vue ne perce le relief plus qu'une fois, ensuit le point d'intersection est raffiné à l'aide d'une recherche binaire.

Dans les deux cas [46], le prochain paramètre t_{i+1} est donné par :

$$t_{i+1} = t_i + \frac{cone_rayon(D[P_i]) - P_{i_z}}{D[P_i]\|v_{xy}\| + cone_rayon} \tag{4.8}$$

Avec $D[P_i]$ représente la profondeur extraite de la carte de déplacement au point P_i.

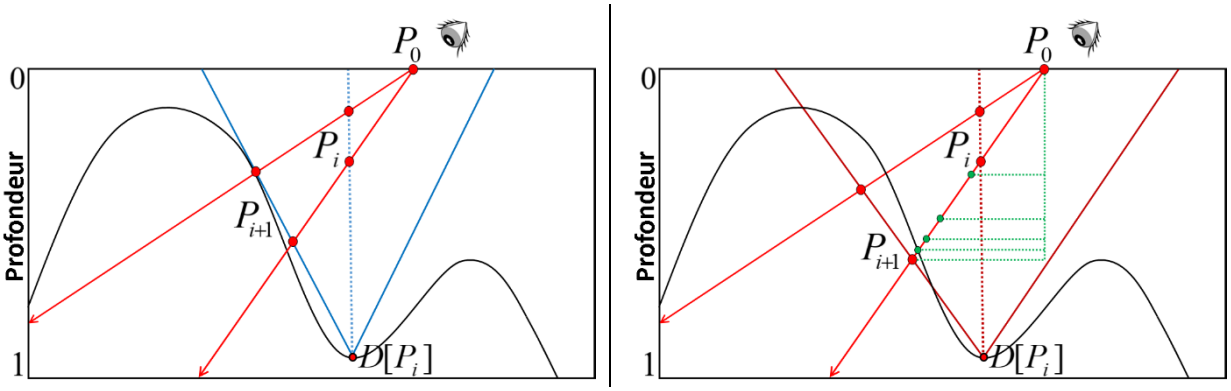


Figure 4.8 : Technique de traçage de cônes. (a) Technique conservative. (b) Technique relaxée.

4.3.2 La technique hybride

Notre idée est de définir un cône hybride de tel sorte que le rayon de vue ne perce le relief plus qu'une fois sans avoir le besoin d'utiliser une recherche binaire tout en gardant une bonne qualité de rendu. Le cône hybride (Figure 4.9) est un cône qui se situe entre le cône conservatif et le cône relaxé, ça veut dire que le rayon d'un cône hybride doit être supérieur à celui du cône conservatif et inferieur à, celui du cône relaxé.

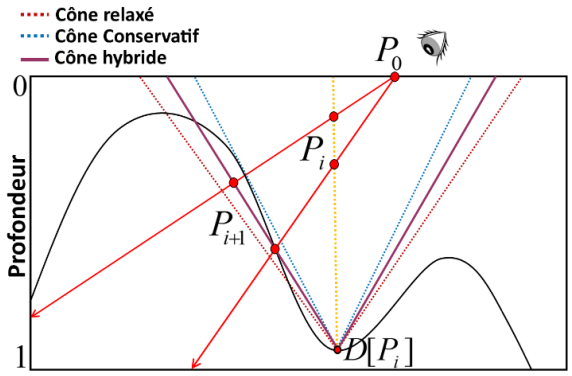
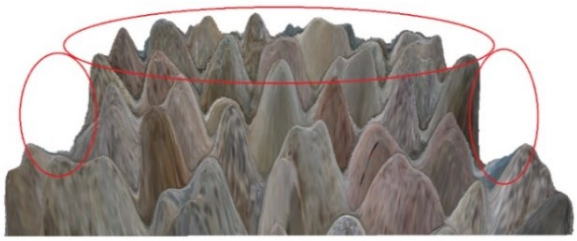


Figure 4.9 : Traçage de cônes hybrides.

L'intérêt d'utiliser un cône hybride est l'élimination de la recherche binaire utilisée avec la technique relaxée en forçant le rayon de vue à percer le relief au plus une fois pendant la recherche de l'intersection. La Figure 4.10 montre une comparaison entre les trois techniques de traçage de cônes. Le rayon d'un cône hybride est calculé dans la phase de rendu à l'aide de la relation suivante :

$$cone_rayon_H = cone_rayon_C + \frac{(cone_rayon_R - cone_rayon_C)}{2} \tag{4.9}$$

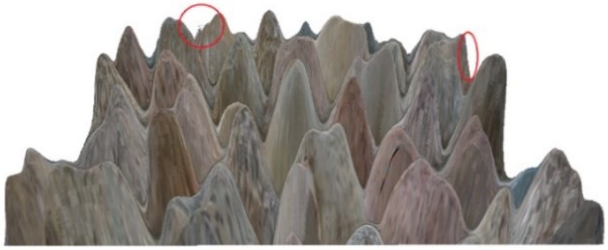
Avec $cone_rayon_C$, $cone_rayon_R$ et $cone_rayon_H$ sont respectivement les rayons des cônes conservatif, relaxé et hybride.



Technique conservative



Technique Relaxée



Technique hybride

Figure 4.10 : Comparaison entre les trois techniques de traçage de cônes.

4.4 TECHNIQUE DE TRAÇAGE DE CONES AVEC PRIS EN CHARGE DE LA SILHOUETTE

La technique de traçage de cônes (dans ses trois versions) reste efficace pour le rendu temps réel sur des surfaces planes, mais lorsqu'il s'agit des surfaces courbées, la silhouette n'est pas visible aux bords des objets 3D (Figure 4.11). Pour résoudre ce problème, nous proposons d'utiliser l'approximation quadratique pour exploiter ses paramètres dans la phase de traçage de cônes afin de déterminer les fragments appartenant à la silhouette.

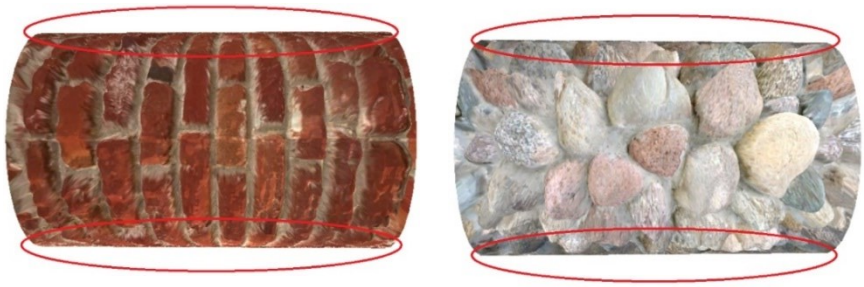


Figure 4.11 : Cylindre rendu avec les techniques originales de traçage de cônes. La silhouette n'est pas visible sur les bords des objets 3D.

Pour mieux comprendre le problème de la silhouette, nous avons deux surfaces, l'une est plane et l'autre est courbée (Figure 4.12a). Pour les surfaces planes, le rayon de vue perce toujours le relief (rayon A), c'est-à-dire qu'on aura toujours une intersection. Mais pour les surfaces courbées, parfois le rayon de vue ne perce pas le relief (rayon B), c'est-à-dire qu'il n'y a pas d'intersection et dans ce cas le pixel appartenant à la silhouette. Alors, l'algorithme de traçage de rayons ne trouvera pas le point d'intersection ou il en trouvera un erroné, et dans ce cas, le pixel doit être ignoré pour avoir un traitement correct de la silhouette.

La Figure 4.12 montre une comparaison d'un cylindre rendu avec la technique de traçage de cônes sans et avec correction de la silhouette et en mettant en évidence le maillage polygonal. On remarque que la silhouette n'est pas visible sur l'objet 3D lorsqu'il est rendu sans correction de la silhouette (Figure 4.12c), ce problème est surmonté en exploitant les paramètres de la surface quadratique pour prendre en compte la courbure de l'objet 3D (Figure 4.12d, Figure 4.12e).

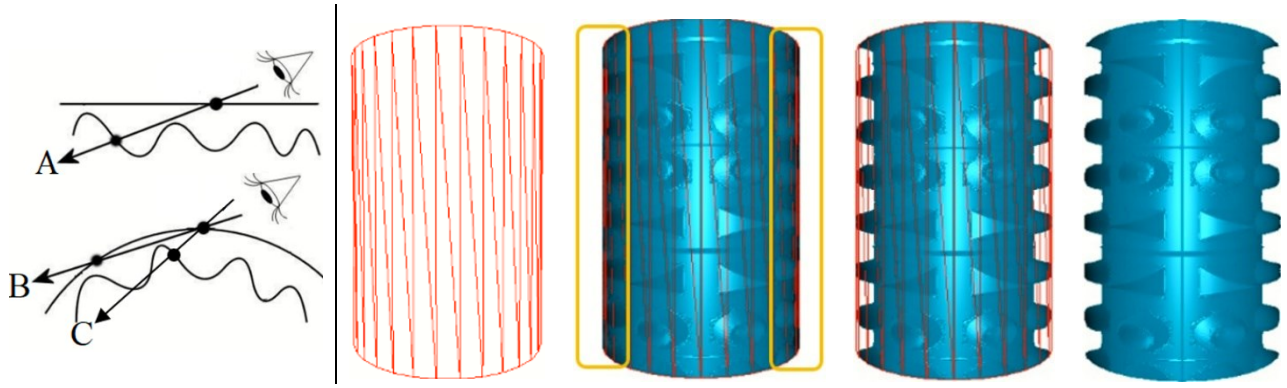


Figure 4.12 : Problème de la silhouette. (a) Une surface plane permet d'avoir une intersection contrairement à une surface courbée. (b) Maillage polygonal. (c) Sans correction de la silhouette. (d)-(e) Avec correction de la silhouette.

4.4.1 L'approximation quadratique

L'approximation quadratique a été utilisé avec la technique du placage de relief [84], elle consiste à calculer une surface quadratique approximative pour chaque sommet du maillage polygonal dans une étape de prétraitement, et durant la phase de rendu, cette surface est utilisée pour adapter le traçage de rayons afin qu'il prenne en compte la forme de la géométrie 3D au niveau du sommet en question.

La surface quadratique approximative est représentée par deux paramètres a et b de tel sorte que :

$$z = ax^2 + by^2 \tag{4.10}$$

(x, y, z) sont les coordonnées du sommet en question.

Ces paramètres sont calculés dans l'espace tangent associé au sommet (x, y, z) en utilisant les quadriques [83], [84]. Soit T l'ensemble de triangles partageons un sommet $p_k(x_k, y_k, z_k)$, et soit $P = \{p_1, p_2, \dots, p_n\}$ l'ensemble de sommets dans T. Tous les sommets dans P sont exprimés dans l'espace tangent de p_k. Etant donné $P' = \{p'_1, p'_2, \dots, p'_n\}$, avec $p'_i = (x'_i, y'_i, z'_i) = (x_i - x_k, y_i - y_k, z_i - z_k)$, les coefficients a et b sont obtenus en résolvant le système $Ax = b$ suivant :

$$\begin{bmatrix} x'^2_1 & y'^2_1 \\ x'^2_2 & y'^2_2 \\ \vdots & \vdots \\ x'^2_n & y'^2_n \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} z'^2_1 \\ z'^2_2 \\ \vdots \\ z'^2_n \end{bmatrix} \tag{4.11}$$

Durant la phase de rendu, les coefficients a et b seront interpolés pour chaque pixel, puis utilisés pour calculer la distance entre le rayon de vue et la surface quadratique. Cette distance est calculée dans l'espace tangent associé au sommet p_k, alors on a deux cas (Figure 4.13).

Soit R un point appartenant à la quadrique Q, $\vec{U}$ est le vecteur unitaire perpendiculaire au $\vec{V}$ au point P, $\vec{V}$ est le rayon de vue exprimé dans l'espace tangent.

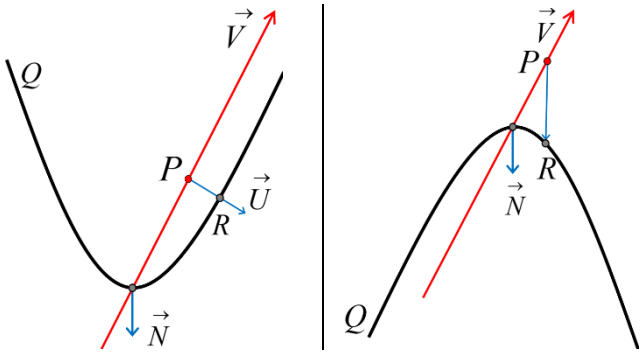


Figure 4.13 : Coupe de deux surfaces quadratiques. La distance entre le rayon de vue est la quadrique est donnée par le segment PR.

Dans le premier cas, $\vec{V}$ est à l'intérieur de la quadrique (Figure 4.13a), R est obtenu en faisant translater P de d unités suivant $\vec{U}$:

$$R = P + \vec{U}d \tag{4.12}$$

La distance entre le point P et la quadrique est simplement d, qui peut être obtenu en substituant les coordonnées de R dans l'équation de la quadrique :

$$aR_x^2 + bR_y^2 - R_z = 0 \tag{4.13}$$

$$a(P_x + dU_x)^2 + b\left(P_y + dU_y\right)^2 - (P_z + dU_z) = 0 \tag{4.14}$$

La résolution de cette équation donne :

$$d = \frac{-B + \sqrt{\Delta}}{2A} = \frac{-B + \sqrt{B^2 - 4AC}}{2A} \tag{4.15}$$

Avec : $$\Delta > 0 \;\; et \;\; \begin{cases} A = \left(aU_x^2 + bU_y^2\right) \\ B = \left(2aP_xU_x + 2bP_yU_y - U_z\right) \\ C = \left(aP_x^2 + bP_y^2 - P_z\right) \end{cases} \tag{4.16}$$

Dans le deuxième cas $\Delta < 0, \vec{V}$ est à l'extérieur de la quadrique (Figure 4.13b), dans ce cas la distance d est :

$$d = P_z - \left(aP_x^2 + bP_y^2\right) = P_z - q \tag{4.17}$$

On note par q la quadrique $\left(aP_x^2 + bP_y^2\right)$ associé aux paramètres a et b.

L'espace de texture est planaire, et durant la phase de rendu, la surface approximative est utilisée pour que cette espace puisse s'adapter à la géométrie de la surface de l'objet 3D au niveau du sommet en question. En réalité, l'espace de texture reste toujours planaire, et pendant la recherche de l'intersection, le rayon de vue est rectifié pour corriger la position du point P_{i+1} en utilisant les caractéristiques de la surface approximative.

Dans le premier cas (Figure 4.13a) le prochain point P_{i+1} est donné par :

$$P_{i+1} = (v + ud)t_{i+1} = (v + w)t_{i+1} \tag{4.18}$$

Avec u est le vecteur $\vec{U}$ normalisé exprimé dans l'espace de texture. On note par w le vecteur quadratique.

Et dans le deuxième cas (Figure 4.13b) le prochain point P_{i+1} est donné par :

$$P_{i+1} = \left(v_x t_{i+1}, v_y t_{i+1}, v_z t_{i+1} - t_{i+1}^2 d\right) \tag{4.19}$$

Dans ce cas, la distance d doit être exprimée dans l'espace de texture.

Puisque la profondeur de v est normalisée (v/v_z), alors dans le premier cas, la distance d doit être divisée par v_z, et dans le deuxième cas elle doit être divisée par v_z^2 , et cela avant la normalisation de la profondeur de v. La Figure 4.14 montre l'allure générale du rayon de vue pendant la recherche de l'intersection.

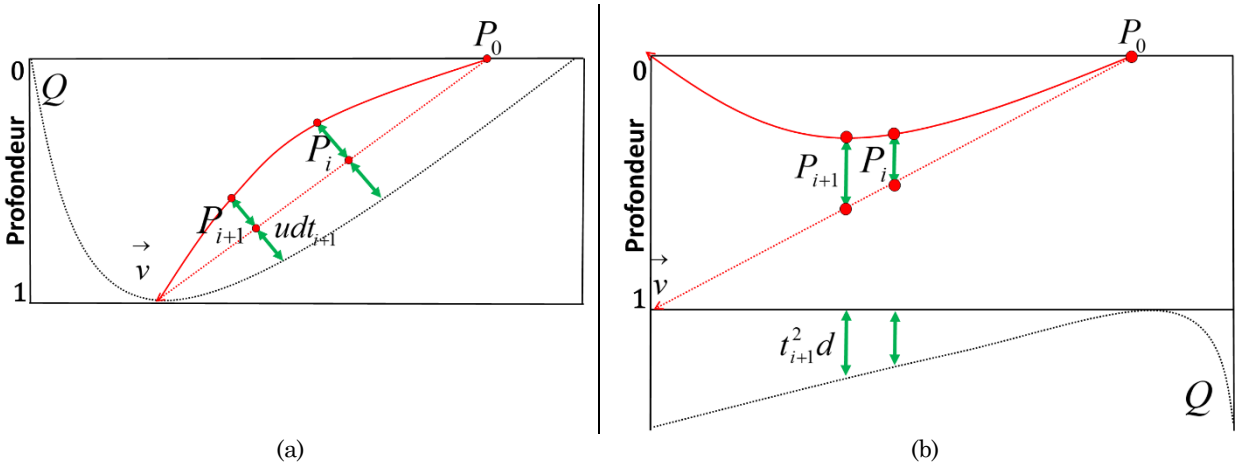


Figure 4.14 : (a) A chaque itération on se rapproche vers la surface quadratique. (b) A chaque itération, on s'éloigne de la surface quadratique.

Pendant la recherche linéaire, la technique du placage de relief originale [59] choisi le paramètre t dans l'intervalle [0,1], c'est-à-dire le *pas* de déplacement est $1/nb$. Cette recherche est optimisée dans [84] en choisissant le paramètre t dans l'intervalle $[0, t_{max}]$. t_{max} est le plus petit $t > 0$ tel que la distance du rayon de vue à la quadrique est égale à 0 ou

1 (Figure 4.15). Pour trouver la valeur la plus précise, t_{max} doit être calculée en substituant (V_x, V_y, V_z) par $(V_x t, V_y t, V_z t)$ et en mettant $d = 0$ et $d = 1$ respectivement dans les deux équations (4.14) et (4.17), puis résoudre pour t. Les algorithmes Algorithme 4.1 et Algorithme 4.2 implémentent cette optimisation.

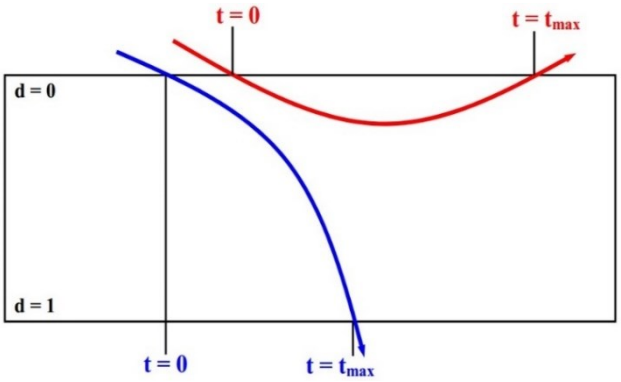


Figure 4.15 : Un rayon qui atteint la profondeur 1 a atteint le fond de la carte de profondeurs caractérisant une intersection (rayon bleu). D'autre part, un rayon qui retourne à la profondeur 0 peut être écarté en toute sécurité (rayon rouge).

Dans la phase de la recherche de l'intersection, le paramètre t_{i+1} est calculé par :

$$t_{i+1} = t_i + \frac{t_{max}}{nb} \tag{4.20}$$

A la fin de la recherche linéaire, on vérifie si la valeur du paramètre t est supérieure à t_{max}, si tel est le cas le pixel doit être ignoré, sinon, on raffine le point d'intersection avec une recherche binaire.

La combinaison du placage de relief avec l'approximation quadratique produit des résultats satisfaisants, mais lorsque l'échelle de profondeur est assez grande ou l'axe de vue rase la surface des artefacts deviennent nettement visible comme montre la Figure 4.16.

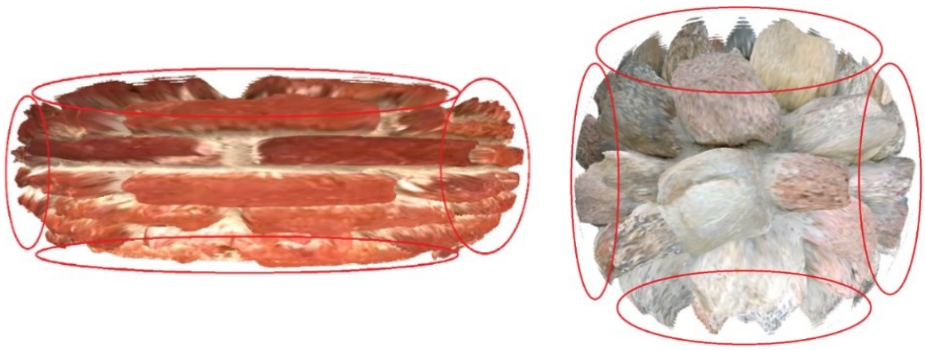


Figure 4.16 : Les défauts sont visibles dans les parties où le rayon de vue rase la surface 3D.

Algorithme 4.1 (tMax1) : Calcul de t_{max} dans le cas où le rayon de vue est à l'intérieur de la quadrique.

```
Entrées : V, U, a, b
Sortie : tmax
Début
 A ← a*V.x*V.x + a*V.y*V.y
 B ← 2*a*V.x*U.x + 2*b*V.y*U.y - V.z
 C ← a*U.x*U.x + b*U.y*U.y - U.z
 D ← B*B - 4*A*C
 Si D > 0 Alors
  tmax ← (B - Racine(D))/(-2*A)
 FinSi
 D ← V.z/A
```

```
  Si D > 0 Alors
   tmax ← Min(tmax,D)
  FinSi
  tmax ← Abs(tmax)
Fin
```

Algorithme 4.2 (tMax2) : Calcul de t_{max} dans le cas où le rayon de vue est à l'extérieur de la quadrique.

```
Entrées : V, q
Sortie : tmax
Début
 D ← V.z*V.z - 4*q
 Si D > 0 Alors
  tmax ← (-V.z + Racine(D))/(-2*q)
 FinSi
 D ← V.z/q
 Si D > 0 Alors
  tmax ← Min(tmax,D)
 FinSi
 tmax ← Abs(tmax)
Fin
```

4.4.2 Traçage de cônes avec correction de la silhouette

Pour corriger le problème de la silhouette, nous allons utiliser les paramètres de la surface quadratique afin d'adapter le processus de traçage de cônes pour qu'il prend en compte les caractéristiques de la surface 3D. Pour cela, nous avons opté pour deux solutions, la première utilise une rectification du rayon de vue après chaque nouveau déplacement le long du rayon de vue et la deuxième utilise une rectification des cônes avant chaque nouveau déplacement.

4.4.2.1 Rectification du rayon de vue

Cette rectification consiste à adapter l'allure du rayon de vue pour qu'il prenne en compte les formes des surfaces quadratiques présentées dans la Figure 4.13. Dans les deux cas, la valeur t_{i+1} est calculée avec la relation (4.8). Dans le premier cas (Figure 4.13a), le point P_{i+1} se rapproche de la surface quadratique, c'est-à-dire dans le cas d'une intersection, on converge rapidement vers la profondeur 1 (Figure 4.17a), dans ce cas, la rectification est effectuée avec la relation (4.18). Et dans le deuxième cas (Figure 4.13b), le point P_{i+1} s'éloigne de la surface quadratique, c'est-à-dire qu'on s'éloigne de la profondeur 1. Dans le cas où il n'y a pas d'intersection, on converge rapidement vers la profondeur 0 (Figure 4.17b). La rectification est effectuée avec la relation (4.19). A la fin de la phase du traçage de cônes, on vérifie si $t > t_{max}$, si tel est le cas, le pixel est ignoré. Les algorithmes Algorithme 4.3 et Algorithme 4.4 implémentent cette rectification. La Figure 4.18 montre une comparaison entre une sphère rendue sans et avec correction de silhouette et la Figure 4.19 montre un tore rendu avec cette approche et en mettant en évidence le maillage polygonal. Il faut noter que le changement de la position de la caméra n'influence pas le processus de rectification ni la qualité de rendu, puisque la rectification est réalisée en temps réel et à chaque mouvement de la caméra, nous aurons une nouvelle image rendue avec une nouvelle rectification.

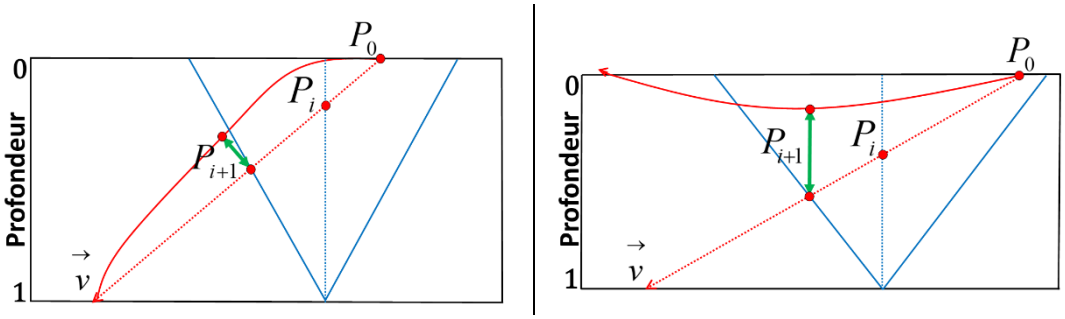


Figure 4.17 : Rectification du rayon de vue. (a) Le point P_{i+1} est rectifié selon la valeur dut_{i+1}. (b) Le point P_{i+1} est rectifié selon la valeur $-t_{i+1}^2 d$.

Algorithme 4.3 (RectRay1) : Traçage de cônes avec rectification du rayon de vue dans le cas où celui-ci est à l'intérieur de la quadrique.

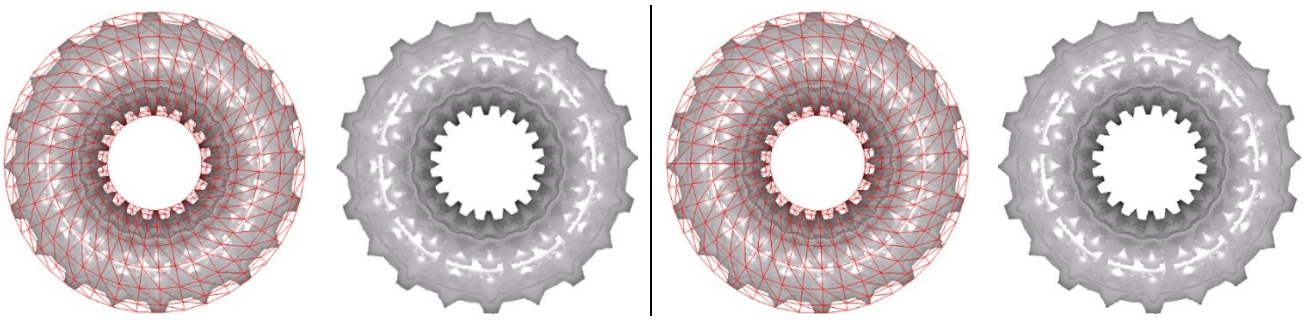
```
Entrée : p0, v, w, tmax, Cd (carte de déplacement)
Sortie : t
Début
 t ← 0
 Pour i = 1 à nb Et t <= tmax Faire
  //Rectification du rayon de vue
  p ← p0 + (v + w)*t
  //Rayon du cône à partir du canal bleu
  cone ← Cd[p.x,p.y].bleu
  //Profondeur du cône à partir du canal alpha
  f ← Cd[p.x,p.y].alpha
  t ← t + (cone * Max(f - p.z, 0)/(cone + f*Norme(v.xy))
 FinPour
Fin
```

Algorithme 4.4 (RectRay2) : Traçage de cônes avec rectification du rayon de vue dans le cas où celui-ci est à l'extérieur de la quadrique.

```
Entrée : p0, v, q, tmax, Cd (carte de déplacement)
Sortie : t
Début
 t ← 0
 Pour i = 1 à nb Et t <= tmax Faire
  p ← p0 + v*t
  //Rectification du rayon de vue
  p.z ← p.z - t*t*q
  //Rayon du cône à partir du canal bleu
  cone ← Cd[p.x,p.y].bleu
  //Profondeur du cône à partir du canal alpha
  f ← Cd[p.x,p.y].alpha
  t ← t + (cone * Max(f - p.z,0)/(cone + f*Norme(v.xy))
 FinPour
Fin
```



Figure 4.18 : Comparaison d'une sphère rendue par les techniques originales de traçage de cônes et par l'utilisation de la rectification du rayon de vue. (a) Technique conservative. (b) Technique relaxée.



Technique conservative Technique relaxée

Figure 4.19 : Un tore rendu en utilisant la rectification du rayon de vue lors du traçage de cônes.

4.4.2.2 Rectification des cônes

La surface approximative a deux formes, concave et convexe (Figure 4.13), alors au lieu d'adapter l'espace de texture à ces fromes, on rectifie les cônes pour qu'ils soient influencés par les caractéristiques de la surface quadratique approximative. Cette rectification est effectuée avant le traçage de cônes et au niveau de ses paramètres, à savoir le rayon et la hauteur (profondeur du cône).

Dans le premier cas (Figure 4.13a), le cône doit s'agrandir pour qu'il s'approche de la surface quadratique, alors la rectification consiste à augmenter les valeurs de ses paramètres en utilisant la distance d et le vecteur u (Figure 4.20a).

Les profondeurs augmentent, ce qui implique l'augmentation de la profondeur (hauteur) du cône, alors la nouvelle profondeur est :

$$D'[P] = D[P] + du_z t \tag{4.21}$$

Et pour le rayon on a :

$$cone_rayon' = cone_rayon \ + d\|u_{xy}\|t \tag{4.22}$$

La Figure 4.20a montre que le déplacement avec le cône rectifié est plus rapide que celui du cône original, car le pas de déplacement augmente. A chaque itération, la rectification du cône fait avancer le point P_{i+1} le long du rayon de vue, et dans le cas d'une intersection, on converge rapidement.

Dans le deuxième cas (Figure 4.13b), le cône diminue pour qu'il s'éloigne de la surface quadratique, alors la rectification consiste à diminuer ses paramètres en utilisant simplement la distance d (Figure 4.20b).

Les profondeurs diminuent, ce qui implique la diminution des profondeurs (hauteurs) des cônes, alors la nouvelle profondeur est :

$$D'[P] = D[P] - t^2 d \tag{4.23}$$

Et on a :

$$cone_ratio = \frac{cone_rayon}{D[P]} = \frac{cone_rayon'}{D'[P]} \tag{4.24}$$

Alors le nouveau rayon du cône est donné par :

$$cone_rayon' = \frac{cone_rayon}{D[P]} D'[P] \tag{4.25}$$

La Figure 4.20b montre que le déplacement avec le cône rectifié est plus lent que celui du cône original, car le pas de déplacement diminue. A chaque itération, la rectification du cône fait reculer le point P_{i+1} le long du rayon de vue, et dans le cas où il n'y a pas d'intersection on diverge.

Dans les deux cas, la valeur t_{i+1} est calculée par :

$$t_{i+1} = t_i + \frac{cone_rayon'\left(D'[P_i] - P_{i_z}\right)}{D'[P_i]\left\|v_{xy}\right\| + cone_rayon'} \tag{4.26}$$

Et le prochain point P_{i+1} est calculé avec la relation (4.2). A la fin du traçage de cônes, on vérifie si $t > t_{max}$, si tel est le cas, le pixel est ignoré. Les algorithmes Algorithme 4.5 et Algorithme 4.6 implémentent cette rectification. La Figure 4.21 montre une comparaison entre une sphère rendue sans et avec correction de silhouette et la Figure 4.22 montre un tore rendu avec cette approche et en mettant en évidence le maillage polygonal.

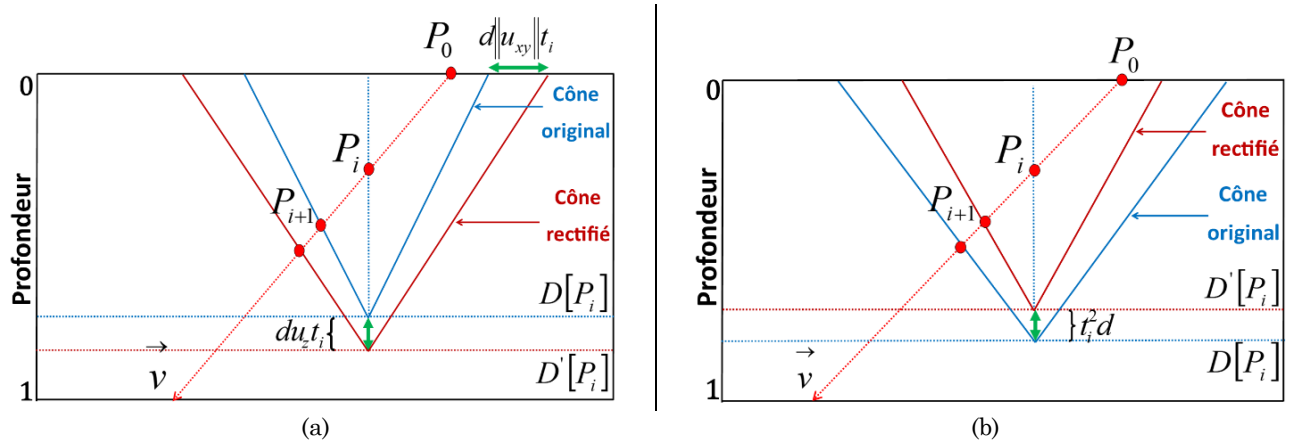


Figure 4.20 : Rectification des cônes. (a) La surface quadratique se rapproche du point P_{i+1}. (b) La surface quadratique s'éloigne du point P_{i+1}.

La rectification des cônes ne dépend pas de la carte de profondeurs, elle ne dépend que des paramètres quadratiques associés à la surface 3D. De plus, la carte de cônes n'est pas attachée à la géométrie de base sur laquelle elle est appliquée, car le processus de rectification est réalisé en temps réel. Cela permet d'utiliser le même processus de rectification et la même texture en temps réel sur différents objets 3D, cela signifie que le processus de rectification et la carte de cônes sont indépendants de la surface sur laquelle ils seront utilisés (Figure 4.23).

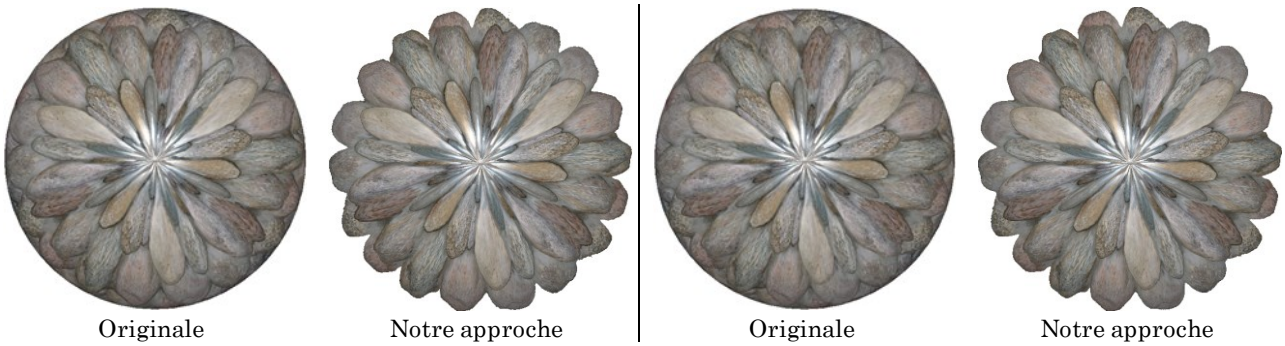
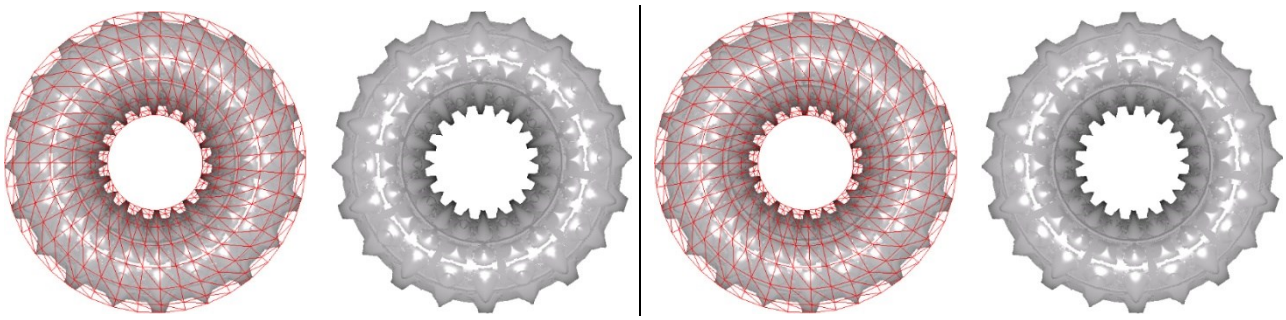


Figure 4.21 : Comparaison d'une sphère rendue par les techniques originales de traçage de cônes et par l'utilisation de la rectification des cônes. (a) Technique conservative. (b) Technique relaxée.



Technique conservative Technique relaxée

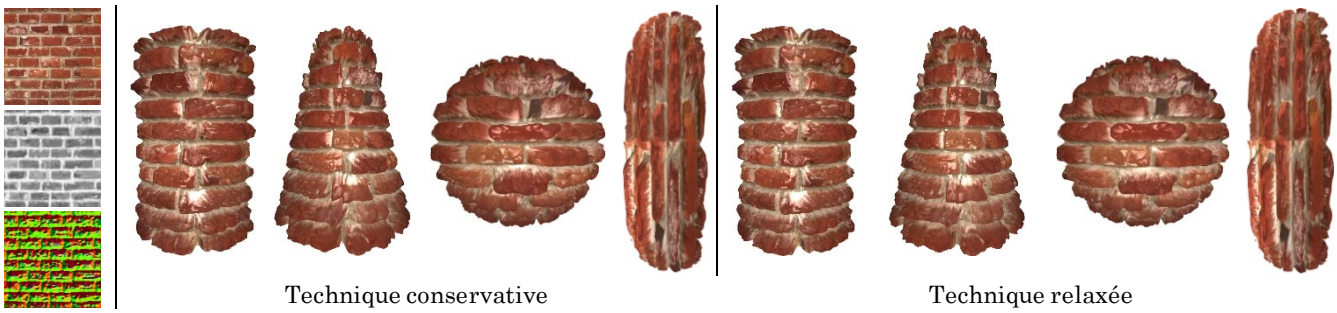
Figure 4.22 : Un tore rendu en utilisant l'approche de rectification des cônes.

Algorithme 4.5 (RectCone1) : Traçage de cônes avec rectification des cônes dans le cas où le rayon de vue est à l'intérieur de la quadrique.

```
Entrée : p0, v, w, tmax, Cd (carte de déplacement)
Sortie : t
Début
 p ← p0 , t ← 0
 Pour i = 1 à nb Et t <= tmax Faire
  cone1 ← Cd[p.x,p.y].bleu
  f1 ← Cd[p.x,p.y].alpha
  //Rectification du rayon du cône
  cone2 ← cone1 + t*Norme(w.xy)
  //Rectification de la profondeur du cône
  f2 ← f1 + t*w.z
  t ← t + (cone2 * Max(f2 - p.z,0)/(cone2 + f2*Norme(v.xy))
  p ← p0 + v*t
 FinPour
Fin
```

Algorithme 4.6 (RectCone2) : Traçage de cônes avec rectification des cônes dans le cas où le rayon de vue est à l'extérieur de la quadrique.

```
Entrée : p0, v, q, tmax, Cd (carte de déplacement)
Sortie : t
Début
 p ← p0 , t ← 0
 Pour i=1 à nb Et t <= tmax Faire
  cone1 ← Cd[p.x,p.y].bleu
  f1 ← Cd[p.x,p.y].alpha
  //Rectification de la profondeur du cône
  f2 ← f1 - t*t*q
  //Rectification du rayon du cône
  cone2 ← (cone1/f1)*f2
  t ← t + (cone2 * Max(f2 - p.z,0)/(cone2 + f2*Norme(v.xy))
  p ← p0 + v*t
 FinPour
Fin
```



Technique conservative Technique relaxée

Figure 4.23 : Rendu de plusieurs objets 3D en temps réel avec le même processus de rectification des cônes et avec la même texture.

4.4.2.3 Algorithme de traçage de cônes avec correction de la silhouette

Dans cette sous-section, nous présentons l'implémentation de l'algorithme de traçage de cônes avec correction de silhouette pour les deux approches (Algorithme 4.7). Durant la phase de rendu, la recherche de l'intersection est effectuée dans l'espace de texture, mais les calculs liés à l'approximation quadratique sont effectués dans l'espace tangent où t est égal à 1 de sorte que la distance quadratique est calculée au fur et à mesure que le déplacement le long du rayon de vue progresse. L'organigramme de la Figure 4.24 résume les étapes principales de l'algorithme.

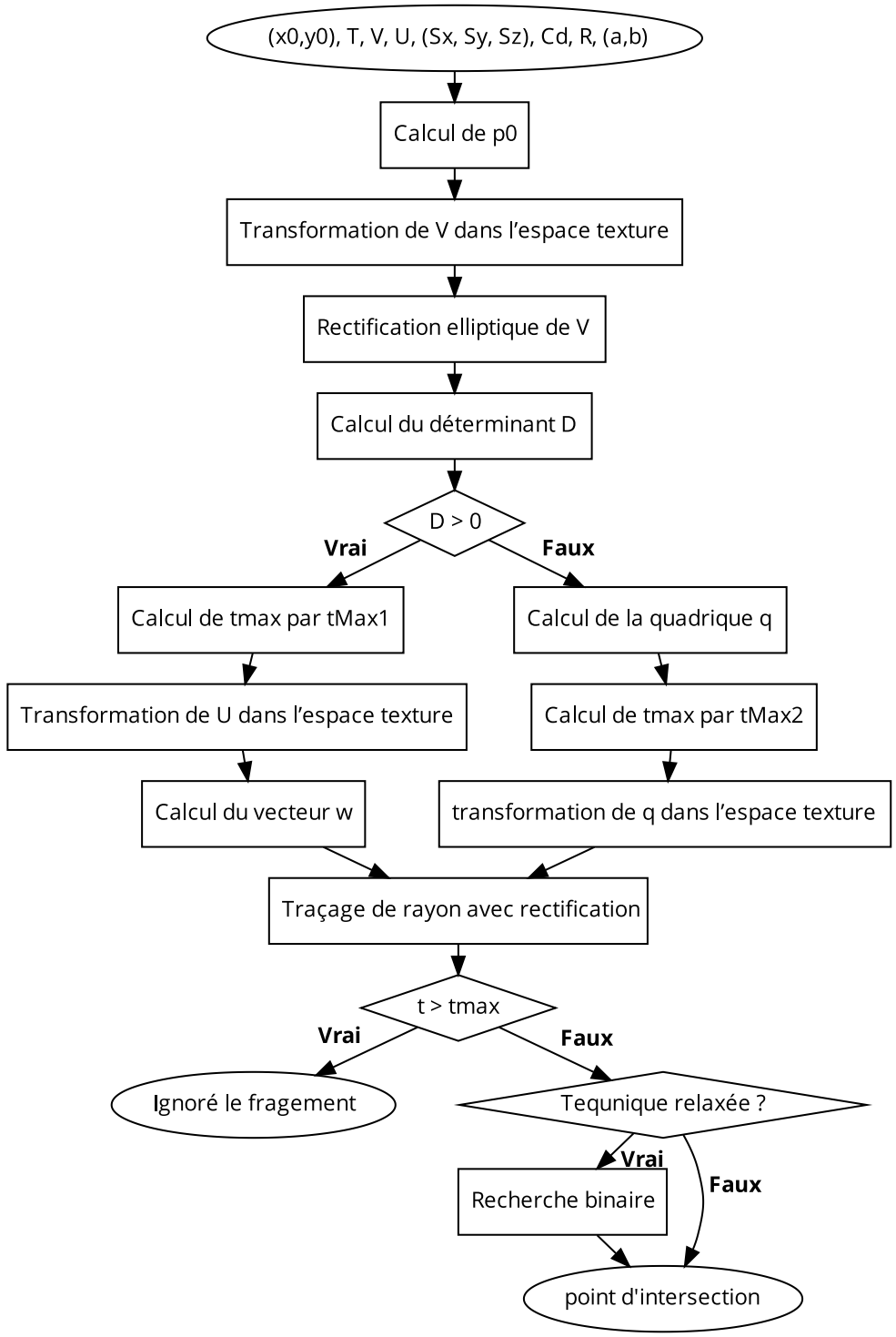


Figure 4.24 : Organigramme de l'algorithme de traçage de cônes avec prise en charge de la silhouette.

Afin d'enlever tout ambiguïté, on note :

- p_0 : Point de départ.
- p : Point d'intersection recherché.
- (x_0, y_0) : Pixel de départ (coordonnées de texture de départ) .
- T : Répétition suivant x et y.
- C_d : Carte de déplacement (ou carte de cônes).
- R : Rapport largeur/hauteur de la carte de déplacement (textures non carrées).
- (S_x, S_y, S_z) : Espace de texture.
- V : Rayon de vue dans l'espace tangent.
- v : Rayon de vue dans l'espace de texture.
- U : Vecteur perpendiculaire au V dans l'espace tangent.
- u : Vecteur U dans l'espace de texture.
- (a, b) : Paramètres de la surface quadratique.
- q : Quadrique.
- w : Vecteur quadratique.
- d : Distance quadratique.

Algorithme 4.7 : Traçage de cônes avec pris en charge de la silhouette.

```
Entrée : (x0,y0), T, V, U, (Sx, Sy, Sz), Cd, R, (a,b)
Sortie : p
Début
 p0 ← (T.x*x0, T.y*x0, 0)
 v ← Normalizer(V/(Sx, Sy, Sz))       //Transformation dans l'espace texture
 v.z ← - v.z
 vz ← v.z
 v ← v/v.z
 //Rectification elliptique pour les textures non carrées
 vR ←(Norme(v.xy)/(Racine(v.x*v.x+R*R*v.y*v.y)
 A ← a*U.x*U.x + b*U.y*U.y
 B ← 2*a*V.x*U.x + 2*b*V.y*U.y - U.z
 C ← a*V.x*V.x + b*V.y*V.y - V.z
 D ← B*B - 4*A*C
 Si D > 0 Alors
  tmax ← tMax1(V, U, a, b))          //Transformation dans l'espace texture
  u ← Normalize(U/(Sx, Sy, Sz))
  w ← ((B - Racine(D))/-2*A)*u/vz
  t←RectRay1(p0, vR, w, tmax ,Cd)     //Rectification du rayon de vue
  //t←RectCone1(p0, vR, w, tmax, Cd) //Rectification des cônes
 Sinon
  q ← a*V.x*V.x + b*V.y*V.y
  tmax ← tMax2(V, q)
  q ← (q/Sz)/(vz*vz)                   //Transformation dans l'espace de texture
  t←RectRay2(p0, vR, q, tmax, Cd)      //Rectification du rayon de vue
  //t←RectCone2(p0, vR, q, tmax Cd)    //Rectification des cônes
 FinSi
 Si t > tmax Alors
  Ignoré le fragment
 Sinon
  p ← p0 + v*t
 FinSi
 // Recherche binaire (uniquement dans le cas des cônes relaxés)
 v ← (v*p.z)/2                        // Taille du pas initial
 p ← p0 + v                           // Point initial
```

```
  Pour i = 1 à nb Faire
   f ← C[p.x,p.y].alpha
   v ← v/2
   Si p.z < f Alors
    p ← p + v
   Sinon
    p ← p – v
   FinSi
  FinPour
Fin
```

4.5 RESULTATS ET DISCUSSION

Les images rendues et les comparaisons entre les différentes techniques discutées dans ce chapitre sont effectuées en utilisant la configuration matérielle *Config.1* (Tableau 0.2).

4.5.1 Synchronisation combinée

Pour comparer les images rendues nous avons utilisé 30 étapes linéaires et 10 étapes binaire dans le cas où il n'y a pas de synchronisation, et dans le cas de la synchronisation, nous avons utilisé l'intervalle [10, 60] pour la recherche linéaire et l'intervalle [2, 8] pour la recherche binaire.

La Figure 4.3 montre l'effet de la synchronisation entre le changement de l'échelle de profondeur et l'ombrage. Les figures Figure 4.4, Figure 4.5 et Figure 4.7 montre l'effet des différentes synchronisations concernant le nombre d'itérations discutées dans cette section. On note qu'une bonne synchronisation est celle qui prend en compte tous les paramètres liés au rendu 3D, à savoir l'ombrage, l'échelle de profondeur et l'ongle de vue.

La synchronisation proposée dans cette section a pour objectif d'optimiser le nombre d'itérations selon la valeur de l'échelle de profondeur et selon l'ongle de vue afin d'obtenir une qualité de rendu satisfaisante. Pour bien profiter de cette synchronisation, il faut bien choisir les valeurs des bornes de l'intervalle $[nb_{min}, nb_{max}]$ pour avoir une vitesse de rendu optimale, ces valeurs doivent être choisies en fonction de la valeur minimale et maximale de l'échelle de profondeur d'une part, et d'autre part en fonction de la valeur minimale et maximale de l'ongle de vue. Cette approche peut être utilisé avec n'importe quel objet 3D et avec n'importe qu'elle technique de placage de déplacement par pixel.

4.5.2 Cônes hybrides

Pour comparer les images rendues avec les différentes techniques, le nombre d'itérations est choisi d'une manière dynamique par rapport à l'échelle de profondeur en utilisant une interpolation avec la relation $x \times (1-k) + y \times k$ avec k représente l'échelle de profondeur qui varie entre 0 et 1, et x et y représentent respectivement le nombre maximal et minimal d'itérations. Pour la recherche linéaire et binaire, on a choisi respectivement les intervalles $[5,25]$ et $[3,7]$.

La Figure 4.10 montre trois images rendues avec les différentes techniques de traçage de cônes avec une échelle de profondeur égale à 0.5. On remarque que la qualité de rendu de la technique hybride s'améliore par rapport aux techniques conservative et relaxée mais avec un cout croisant. Dans certain cas, des défauts deviennent visibles dans la technique relaxée et qui sont corrigés dans la technique hybride comme le montre la figure.

Le fait de calculer le rayon du cône hybride pendant la phase de rendu ralentit un peu sa vitesse car on accède à la carte de cône deux fois. La meilleure façon d'exploiter cette technique est de calculer une carte de cônes hybride dans une étape de prétraitement à partir des deux cartes de cône conservatif et relaxé.

4.5.3 Traitement de la silhouette

Les images des figures sont rendues avec 25 *pas* linéaires et 5 *pas* binaires. Et pour la résolution des textures, nous avons utilisé dans la plupart des cas la résolution 512×512.

Les figures Figure 4.18 et Figure 4.21 montrent des comparaisons d'une sphère rendue sans et avec correction de la silhouette en utilisant les deux rectifications, à savoir la rectification du rayon de vue et la rectification des cônes. Les images de la Figure 4.19 sont rendues avec l'approche de rectification du rayon de vue et celles de la Figure 4.22 sont rendues avec l'approche de rectification des cônes. La Figure 4.23 montre plusieurs objets 3D rendus avec la rectification des cônes en utilisant la même texture.

La Figure 4.25 montre une comparaison entre les techniques originales de traçage de cônes et l'ajout de la rectification du rayon de vue et de la rectification des cônes. La figure montre clairement que les techniques originales souffrent du problème de la silhouette. C'est-à-dire que lors de la recherche du point d'intersection, la technique considère que chaque surface 3D traitée est plane, et ne tient pas compte de sa courbure. Ensuite, les reliefs aux bords des objets 3D sont confondus avec leurs surfaces, ce qui produit un rendu incorrect. Les deux rectifications proposées ont résolu ce problème.

Concernant la comparaison entre le placage de relief avec correction de la silhouette [84] et le traçage de cônes combiné aux rectifications proposées, nous avons constaté que le problème majeur est lié aux angles de vue rasants. La combinaison du traçage de cônes avec l'approximation quadratique a résolu ce problème. La Figure 4.26 montre la disparition des artefacts dans les images rendues par nos rectifications. Le même problème persiste dans le cas de l'interpénétration d'objets 3D.

Les qualités des images des figures Figure 4.18, Figure 4.19, Figure 4.21 et Figure 4.22 sont proches, mais dans certains cas, lorsque l'angle de vue ou l'échelle de profondeur sont modifiés, des petites différences deviennent nettement visibles comme illustré par les figures Figure 4.27 et Figure 4.28 qui montrent une comparaison d'un tore rendu avec les deux rectifications proposées.

Les qualités des images rendues par les deux approches (c'est-à-dire la rectification du rayon de vue et la rectification des cônes) semblent identiques. Des différences minimes peuvent être observées lorsque nous utilisons un nombre minimal d'itérations (c'est-à-dire

des itérations linéaires ≤ 25, et des itérations binaires ≤ 5), mais lorsque le nombre d'itérations est supérieur, les qualités des images se rapprochent.

La Figure 4.29 montre un cylindre rendu avec les différentes approches discutées dans cette section en utilisant deux résolutions de texture : 256×256 et 2048×2048 et deux positions de la caméra (loin et proche). La figure montre que la résolution de texture joue un rôle très important dans la qualité des images rendues, une haute résolution permet d'avoir une meilleure qualité et donc d'éviter les problèmes de crénelage.

La Figure 4.30 montre un tore rendu avec la technique de traçage de cônes avec correction de la silhouette en utilisant différents nombres d'étapes et en mettant en évidence le maillage polygonal. On observe que le nombre d'itérations joue un rôle très important pour le rendu des reliefs et pour le traitement de la silhouette.

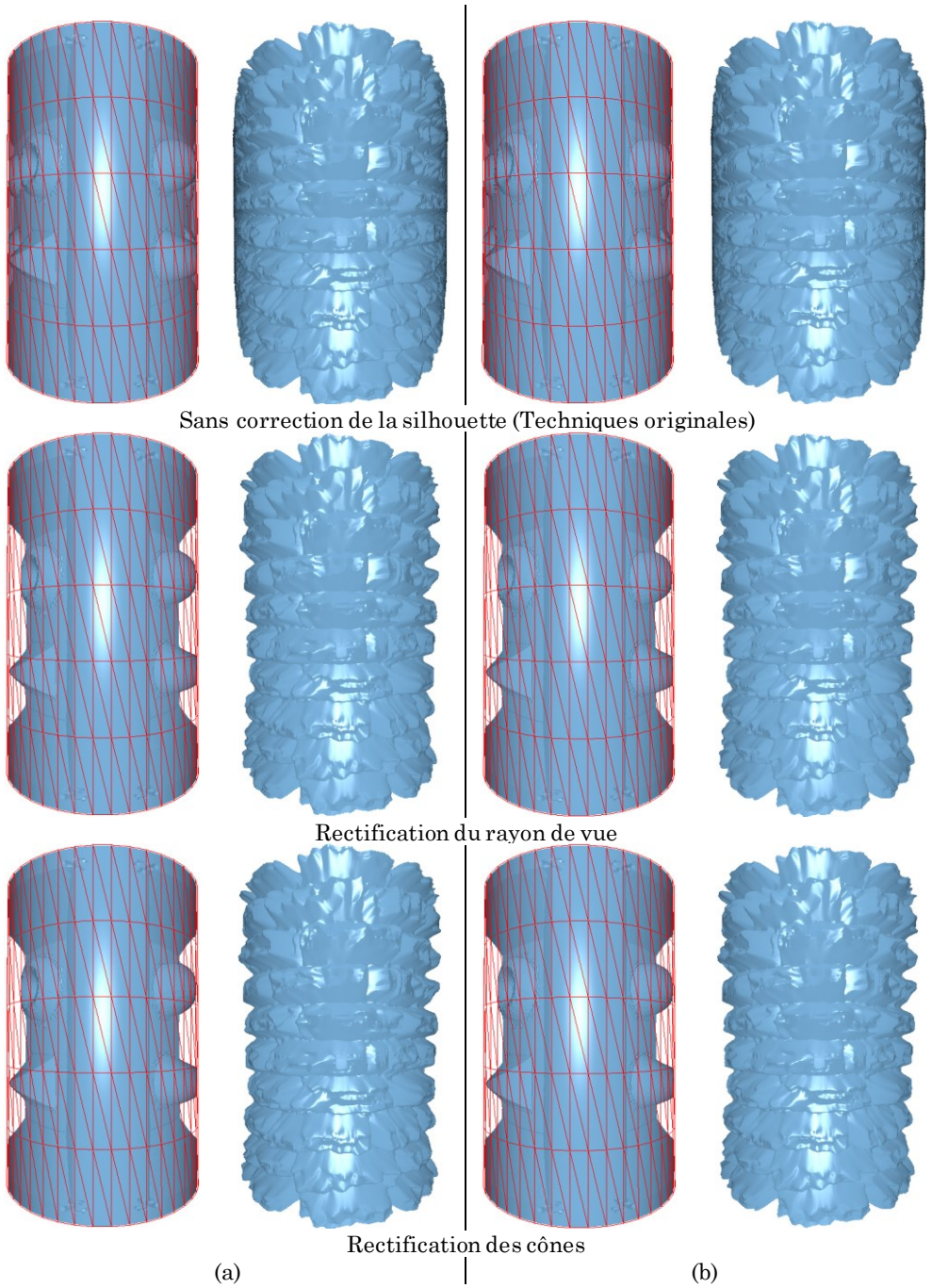


Figure 4.25 : Comparaison d'un cylindre rendu par les techniques originales de traçage de cônes et par l'utilisation des rectifications proposées. (a) Technique conservative. (b) Technique relaxée.

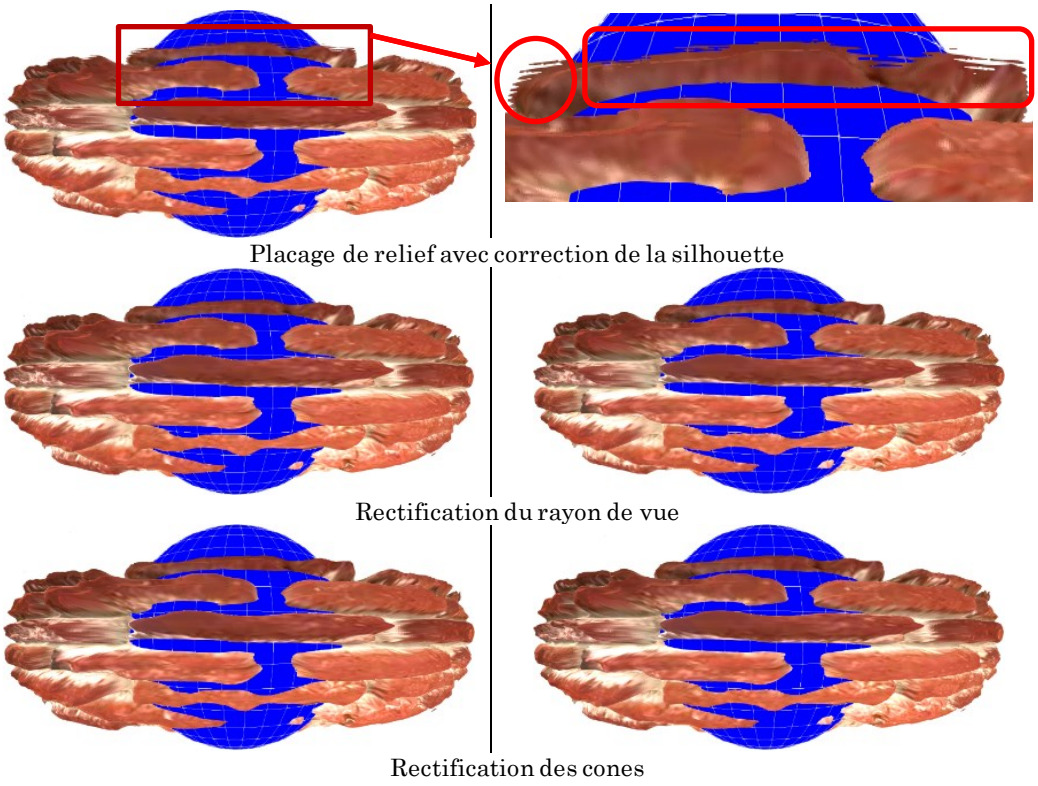


Figure 4.26 : Interpénétration de deux objets 3D, une sphère et un tore. (a) Technique conservative. (b) Technique relaxée.

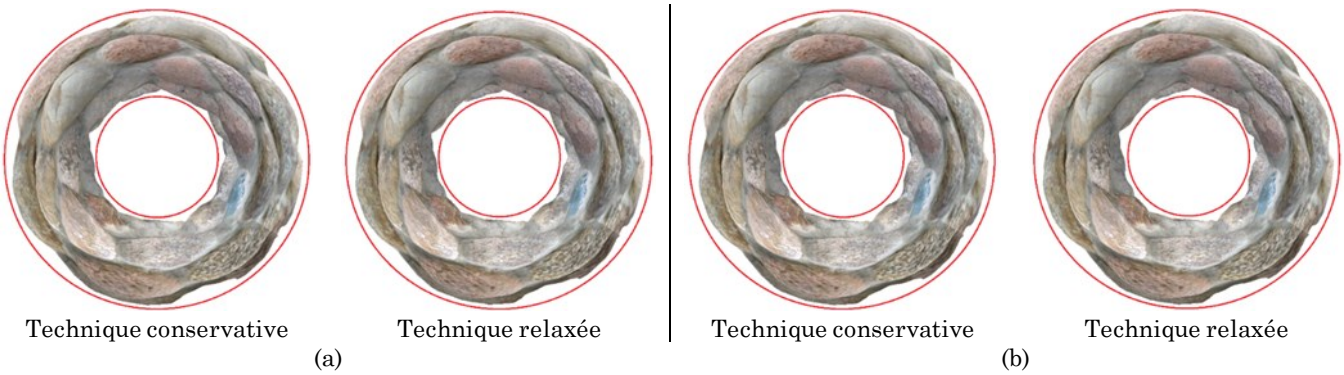


Figure 4.27 : Rendu d'un tore. (a) Rectification du rayon de vue. (b) Rectification des cônes.

La Figure 4.31 montre quelques exemples supplémentaires avec des reliefs à haute fréquence et sans texture de couleur. Les objets 3D sont rendus par la technique de traçage de cônes avec correction de la silhouette et en utilisant différentes cartes et différentes valeurs de profondeur. La Figure 4.32 montre un vase rendu par la technique du traçage de cônes avec correction de la silhouette et en mettant en évidence le maillage polygonal de base. L'objet 3D est rendu avec différentes cartes et différentes valeurs de profondeur. Dans les différentes images de la figure, on remarque que la silhouette est corrigée quelle que soit la carte de profondeurs utilisée.

Généralement, la combinaison de la technique de traçage de cônes avec l'approximation quadratique produit des résultats satisfaisants, mais dans certains cas, cette combinaison produit des distorsions comme le montre la Figure 4.33. Ce problème est dû à l'utilisation de l'approximation quadratique pour la représentation locale de la surface à chaque sommet. Parce que parfois, le rayon de vue perce le relief dans l'espace objet et le quitte dans l'espace texture. Ce problème a également été mentionné dans [85], [86], [91].

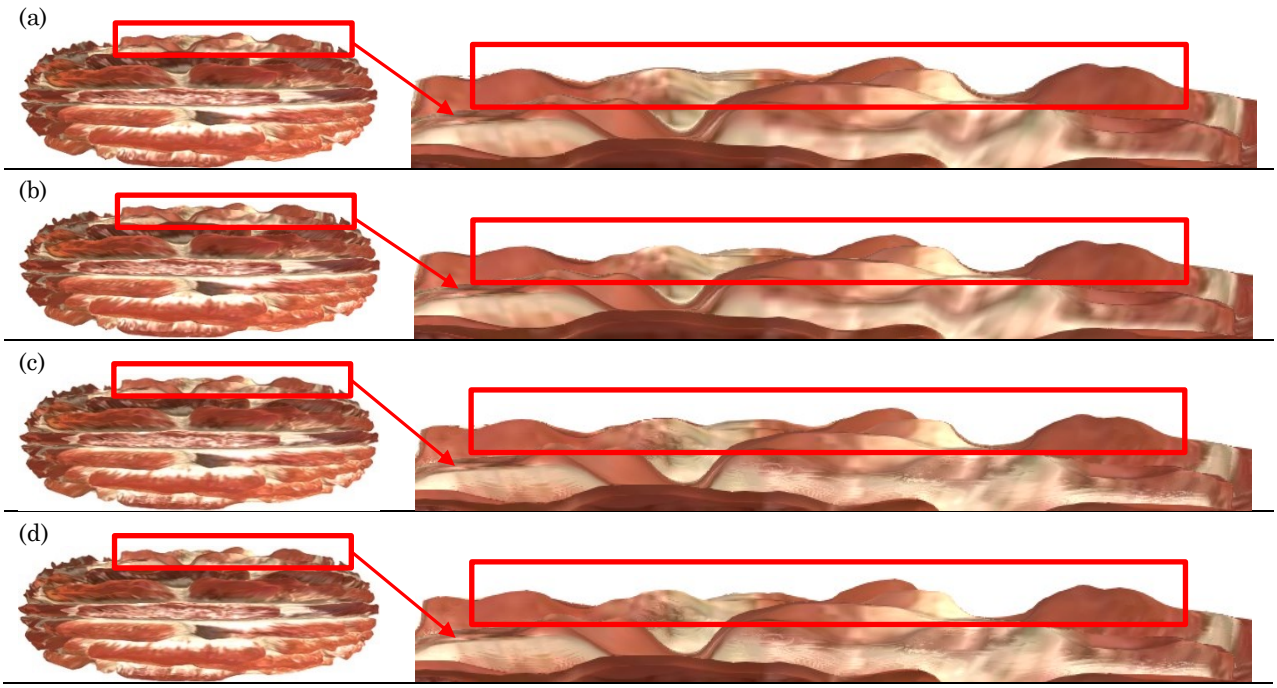


Figure 4.28 : Rendu d'un tore. (a) Technique conservative avec rectification du rayon de vue. (b) Technique conservative avec rectification des cônes. (c) Technique relaxée avec rectification du rayon de vue. (d) Technique relaxée avec rectification des cônes.

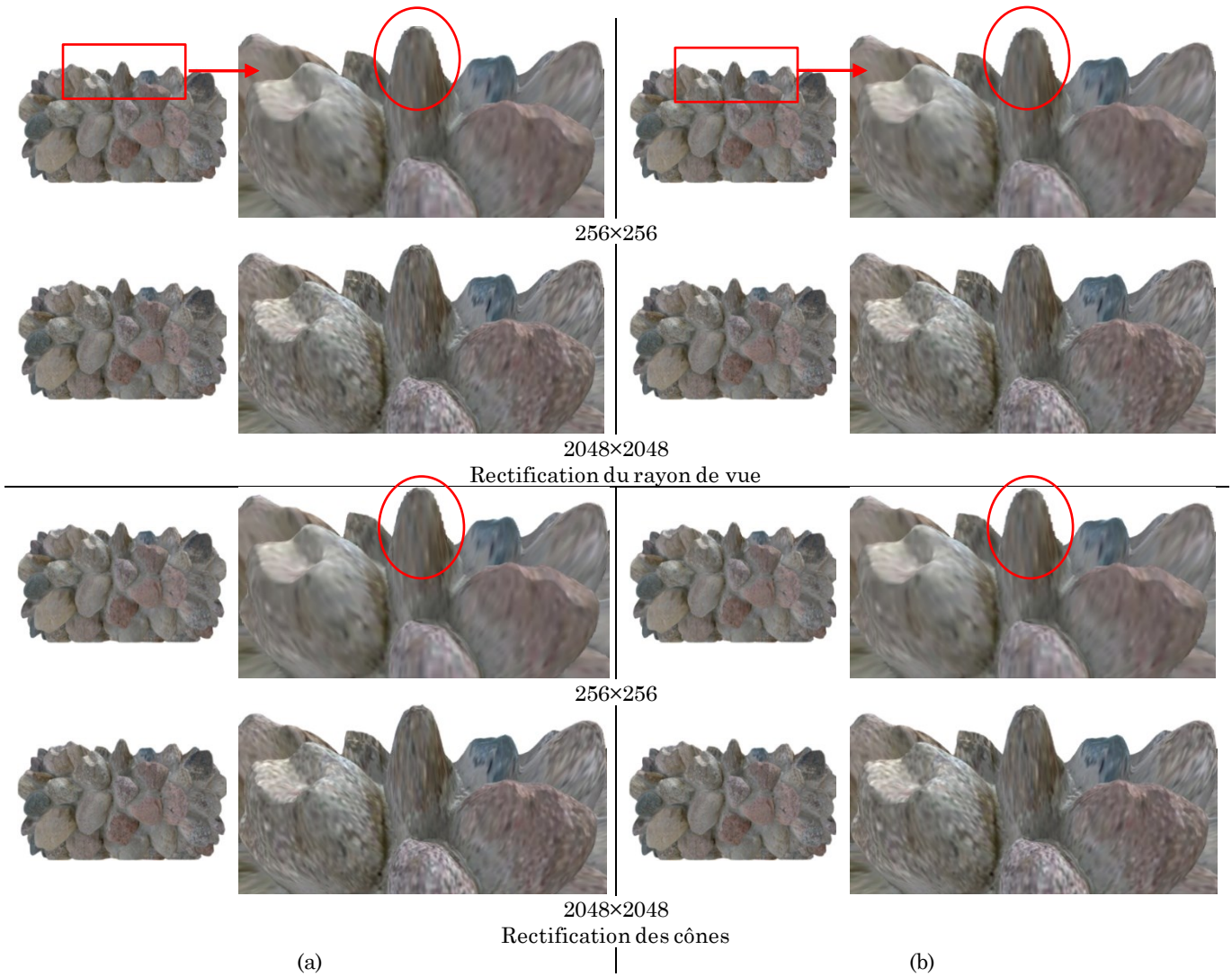


Figure 4.29 : Rendu d'un cylindre avec deux résolutions de texture : 256×256 et 2048×2048. (a) Technique conservative. (b) Technique relaxée.

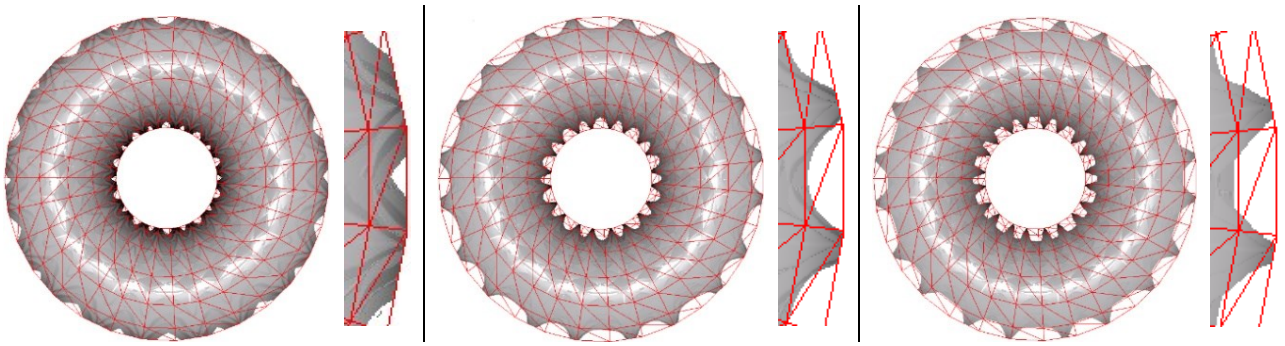


Figure 4.30 : Un tore rendu avec la technique de traçage de cônes avec correction de la silhouette en utilisant un nombre différent d'itérations linéaires (5, 10 et 25).

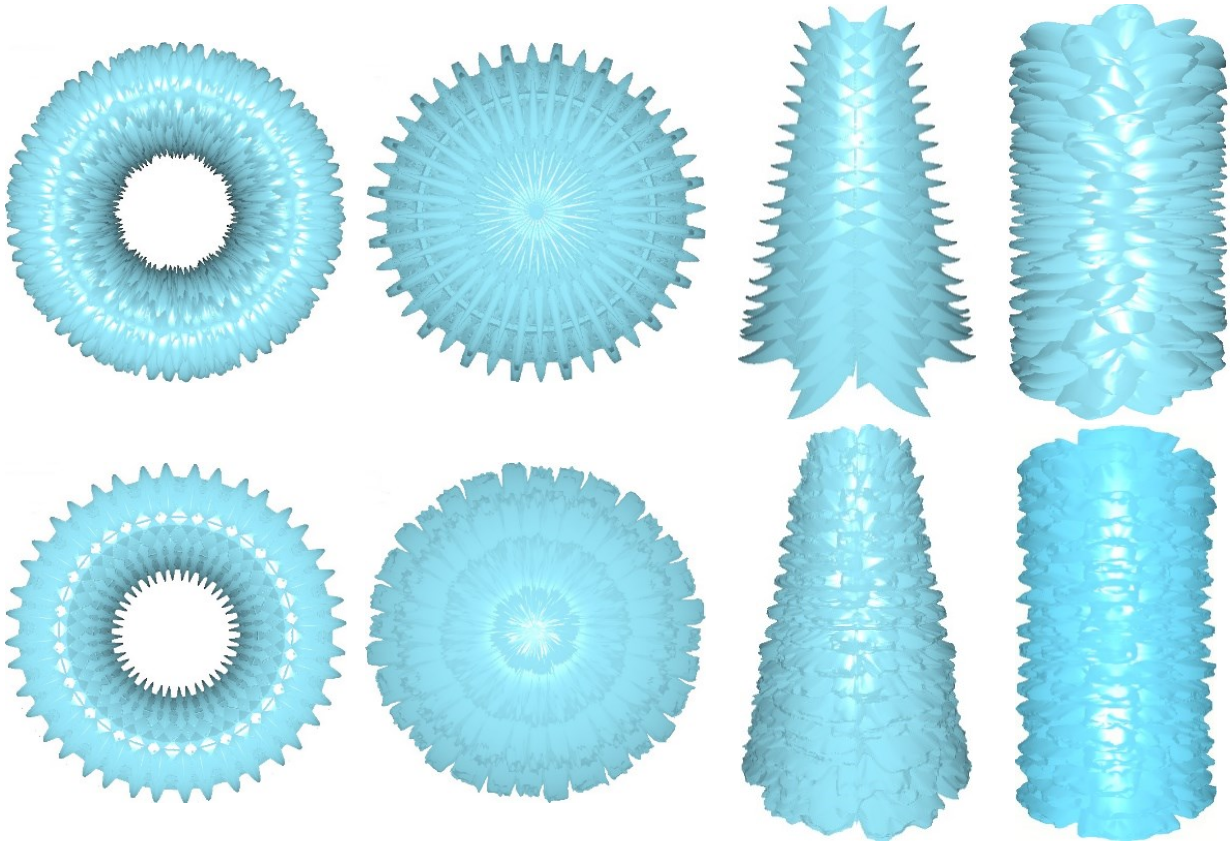


Figure 4.31 : Différents objets 3D rendus par la technique de traçage de cônes avec correction de la silhouette et en utilisant des reliefs à haute fréquence.

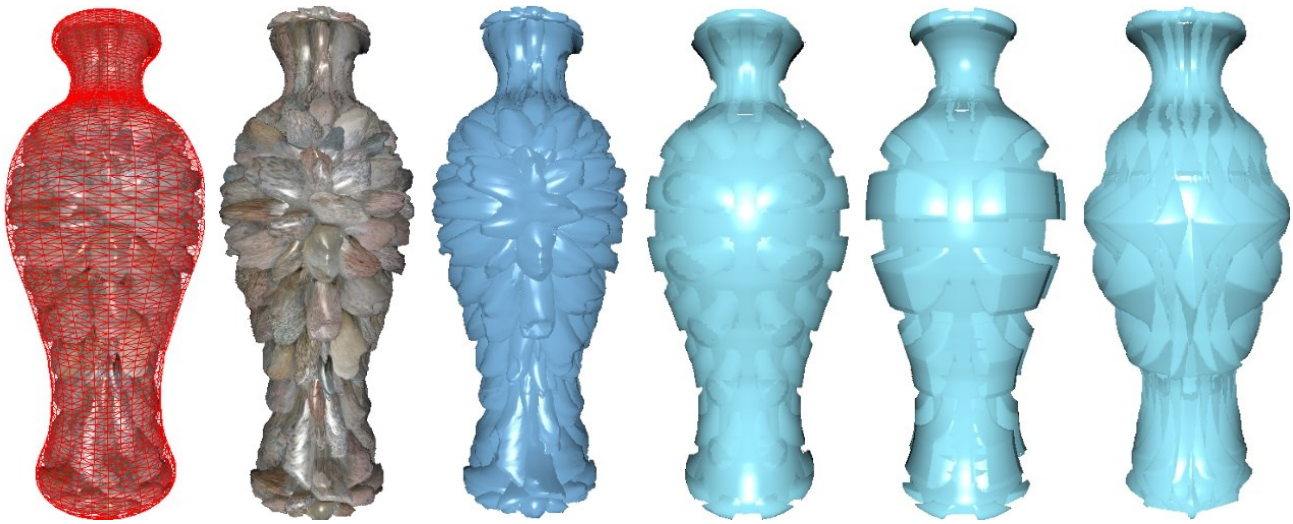


Figure 4.32 : Rendu d'un vase avec la technique de traçage de cônes avec correction de la silhouette en utilisant différentes cartes et différentes valeurs de profondeurs.

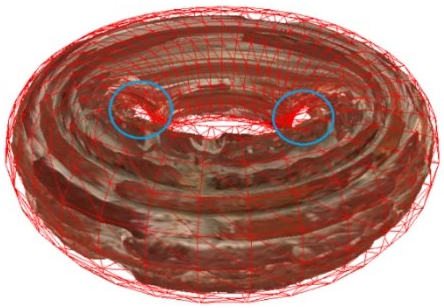
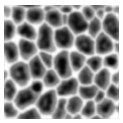


Figure 4.33 : Distorsions dus à l'utilisation de l'approximation quadratique.

Pour comparer la vitesse de rendu, nous avons utilisé des textures de haute résolution 1024×1024 et 2048×2048, 35 itérations pour la recherche linéaires, 10 itérations pour la recherche binaires et une échelle de profondeur égale à 1. Le Tableau 4.1 et la Figure 4.34 montrent la différence de la vitesse de rendu entre les approches discutées dans cette section. Il est clair que l'approche par rectification des cônes est la plus rapide. On note également que la technique relaxée est toujours plus lente que la technique conservative car sa vitesse est ralentie par la recherche binaire.

Tableau 4.1 : Comparaison de la vitesse de rendu IPS (Images Par Seconde) entre les techniques de traçages de cônes et le placage de relief [84].

Scène 1000×600	Texture	Résolution	Technique conservative			Technique relaxée			Placage de relief avec correction de la silhouette [84]
			Rectification du rayon de vue	Rectification des cônes	Sans rectification	Rectification du rayon de vue	Rectification des cônes	Sans rectification	
		1024^2	210	**225**	227	198	**210**	226	202
		2048^2	115	**127**	140	107	**117**	141	105
		1024^2	230	**258**	280	211	**227**	262	197
		2048^2	140	**148**	190	130	**137**	168	117
Moyenne			173	**189**	209	161	**172**	199	155

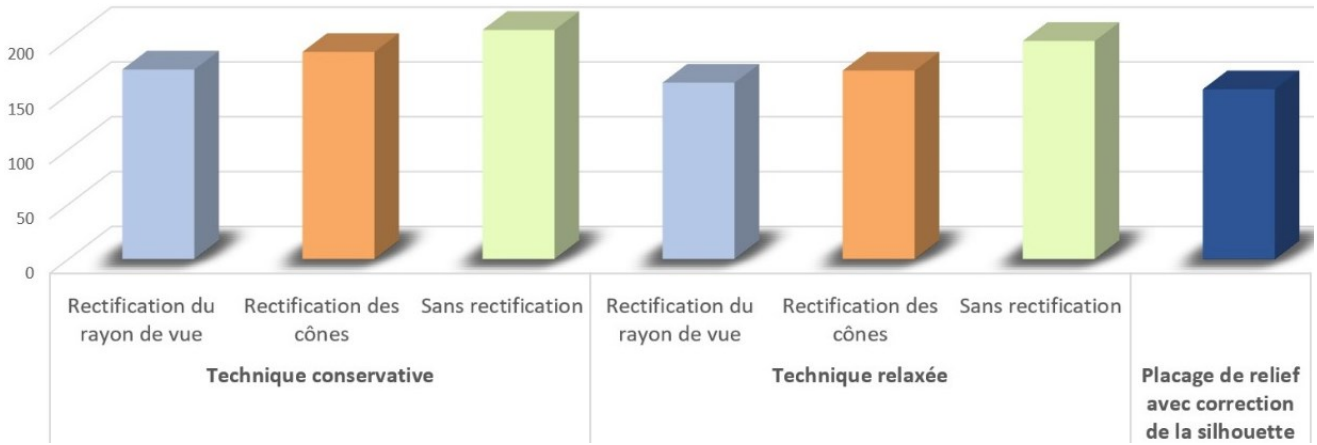


Figure 4.34 : Comparaison de la vitesse moyenne de rendu entre les techniques de traçage de cônes et la technique du placage de relief [84].

Comme le montrent les tableaux Tableau 4.1 et Tableau 4.2, la vitesse de rendu diminue de 37 images par seconde pour la rectification du rayon de vue et de 24 images par seconde pour la rectification des cônes et cela par rapport aux techniques de traçage de cônes originales. Ce ralentissement est dû au traitement concernant la correction de la silhouette, ce qui est bien normale.

Tableau 4.2 : Le nombre moyen de IPS (Images Par Seconde) de la diminution de la vitesse de rendu des deux rectifications proposées par rapport aux techniques originales de traçage de cônes.

Technique	Rectification du rayon de vue	Rectification des cônes
Technique conservative originale	36	20
Technique relaxée originale	38	27

4.6 CONCLUSION

Dans ce chapitre, nous avons présenté trois améliorations concernant le placage de déplacement par pixel, à savoir le placage dynamique de relief, le traçage de cônes hybride et le traçage de cônes avec correction de la silhouette.

Le placage dynamique de relief consiste à utiliser une synchronisation combinée entre le changement de l'échelle de profondeur, le changement de l'ongle de vue et le nombre d'itérations. Cette synchronisation a pour objectif d'optimiser le nombre d'itérations selon la valeur de l'échelle de profondeur et selon l'ongle de vue afin d'obtenir une qualité de rendu satisfaisante.

Le traçage de cônes hybrides consiste à utiliser un cône qui se situe entre le cône conservatif et le cône relaxé, autrement dit, le cône hybride joue le rôle d'interpolateur entre les deux. L'objectif de cette amélioration est d'optimiser le coup de traitement des techniques conservative et relaxée tout en gardant une qualité de rendu acceptable.

Le traçage de cônes avec correction de la silhouette consiste à exploiter l'approximation quadratique afin de déterminer les fragments appartenant à la silhouette. Cette approximation consiste à représenter la surface 3D par des paramètres approximatifs à chaque sommet constituant le maillage polygonal correspondant. Alors, nous avons proposé deux nouveaux algorithmes.

Lors de la phase de traçage de cônes, le premier algorithme consiste à utiliser la surface quadratique approximative pour rectifier le rayon de vue. Cette rectification permet de savoir si le rayon de vue perce ou quitte le relief et il est réalisé après chaque nouveau déplacement le long du rayon de vue. Le deuxième algorithme utilise cette surface approximative pour rectifier les paramètres des cônes (c'est-à-dire la profondeur et le rayon) pour qu'ils soient influencés par ses caractéristiques. Cette rectification permet de savoir si les déplacements le long du rayon de vue permettent d'avoir ou non une intersection et il est réalisé avant chaque nouveau déplacement le long du rayon de vue.

Dans certains cas, les qualités des images rendues par les deux algorithmes sont presque identiques, à l'exception de différences minimes. Cependant, lorsqu'il s'agit de la vitesse de

rendu (nombre d'images par seconde), l'approche par rectification des cônes reste la plus rapide. De plus, la qualité de rendu des microreliefs et de la silhouette dépend de la résolution de la texture et du nombre d'itérations utilisées. Le choix entre la technique conservative et la technique relaxée, la résolution de la texture et le nombre d'itérations peut être réalisé selon le critère *qualité de rendu / vitesse de rendu*.

L'avantage des techniques de traçage de cônes est qu'elles peuvent être combinées avec n'importe quelle approche de traitement de la silhouette avec plusieurs améliorations possibles. En effet, le traitement de la silhouette dépend de l'algorithme de traçage de rayons, et de la façon dont les fragments appartenant à la silhouette sont déterminés. Ainsi, un bon couplage permet d'avoir un meilleur algorithme de traçage de rayons et qui supporte le traitement de la silhouette.

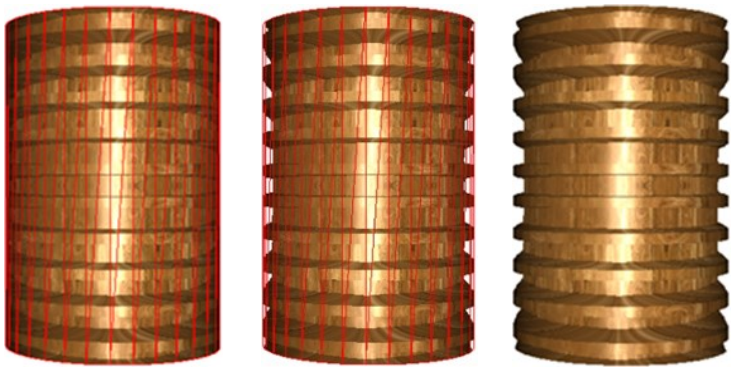
Chapitre 5

Placage d'extrusion par pixel avec traitement de la silhouette

5.1 INTRODUCTION

Le placage d'extrusion par pixel, comme décrit dans [74], [78], se présente comme une méthode ingénieuse dans le domaine de la modélisation géométrique. Cette technique spécifique permet de générer des surfaces en extrudées au-dessus de maillages polygonaux existants, tout en évitant d'accroître leur densité. En outre, cette technique présente des résultats améliorés en évitant les inconvénients généralement associés à une grande échelle de profondeur et aux angles de vue rasants, particulièrement lorsqu'ils sont proches de la surface du modèle. En effet, l'utilisation d'une grande échelle de profondeur peut souvent entraîner des distorsions visuelles indésirables, tandis que les angles de vue rasant la surface 3D peuvent compromettre la qualité de la représentation en créant des discontinuités des surfaces extrudées non souhaités.

Cette technique produit des résultats satisfaisant dans le cas des surfaces planes, mais lorsqu'il s'agit des surfaces courbées, la silhouette n'est pas visible sur aux bords de la géométrie (Figure 5.1). A partir de cette constatation, nous allons présenter dans ce chapitre deux nouveaux algorithmes pour corriger le problème de la silhouette en faisant la combinaison du placage d'extrusion par pixel avec l'approximative quadratique [83], [84].



(a) (b) (c)

Figure 5.1 : Placage d'extrusion par pixel. (a) Placage d'extrusion de base. (b)-(c) Correction de la silhouette.

5.2 PLACAGE D'EXTRUSION PAR PIXEL

Le placage d'extrusion par pixel [74], [78] est une technique de placage de déplacement par pixel [45] qui s'appuie sur des textures qui servent à stocker des données relatives à la géométrie. Cette géométrie est restituée par la suite à l'aide d'un algorithme simplifié de type traçage de rayons qui s'exécute dans les unités programmables de la carte graphique.

Contrairement aux techniques classiques du placage de déplacement par pixel qui se basent sur une image au niveau de gris représentant un relief quelconque, le placage d'extrusion utilise une image binaire 2D, où seuls les pixels à valeur zéro constituent la forme de base du motif qui sera extrudé. Afin converger rapidement vers le point d'intersection entre le rayon de vue et le motif extrudé, cette technique utilise une carte de distance calculée à partir de la forme de base stockée dans une image binaire.

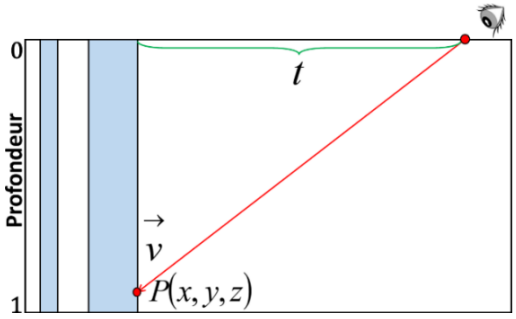


Figure 5.2 : Le premier point d'intersection entre le rayon de vue avec le motif extrudé.

5.2.1 La carte de forme

La carte de forme est une texture RVBA qui contient les données nécessaires aux algorithmes de placage d'extrusion (Figure 5.3). Le canal alpha sert à stocker la forme de base représentée par une image binaire où seuls les pixels à valeur zéro sont considérés comme faisant partie de la forme. Le canal bleu est utilisé pour stocker la carte de distance calculée à partir du canal alpha. Enfin, les canaux rouge et vert abritent respectivement les composantes x et y de la normale calculée à partir de la carte de distance. La carte de distance est calculée en utilisant la transformation de distance euclidienne EDT décrite dans [75]–[77]. Chaque pixel représente la distance minimale séparant celui-ci à la forme extrudée.



(a) (b) (c)

Figure 5.3 : Illustration d'une carte de forme. (a) La forme de base. (b) La carte de distance. (c) La carte de forme.

5.2.2 Le traçage de rayons

Soit (u, v) les coordonnées du pixel courant et p_0 le point de départ de la recherche avec les coordonnées $(x_0, y_0, z_0) = (u, v, 0)$. Et soit v la direction de vue exprimée dans l'espace texture associé au pixel courant. A chaque itération, la distance minimale entre le point P_i et le motif extrudé est extraite du canal bleu de la carte de forme et un traçage horizontal de cercles est effectué pour avancer sans risquer de sauter la première intersection (Figure 5.4).

Pour calculer le prochain point P_{i+1}, on utilise toujours la relation (4.2) et pour calculer le paramètre t_{i+1}, on utilise la relation suivante :

$$t_{i+1} = t_i + D(P_i) \tag{5.1}$$

Avec t_i représente la somme des distances à l'itération i, et $D(P_i)$ représente la distance aux coordonnée (x_i, y_i) séparant le point P_i de la forme extrudée, et qui est calculée à l'aide de la formule suivante :

$$D(P_i) = \tau \times EDT(P_i) \tag{5.2}$$

Où $EDT(P_i)$ représente la transformation de distance euclidienne extraite de la carte de forme aux coordonnées (x_i, y_i), et τ représente le facteur d'échelle [74], [78].

L'utilisation de l'EDT pour sauter l'espace vide pendant le traçage de rayons permet une convergence plus rapide et évite les intersections manquantes aux angles rasants. Étant donné que les champs de distance sont généralement stockés dans un canal au format entier 8 bits, les distances doivent être étalées dans l'intervalle [0, 1] afin d'avoir une distribution optimale. Ainsi, les distances sont modulées en fonction de la valeur maximale de l'EDT et d'une valeur d'échelle f utilisée dans l'étape de prétraitement pour contrôler la distribution des distances, de sorte que le facteur d'échelle est calculé par la formule :

$$\tau = f \frac{max(EDT)}{W} \tag{5.3}$$

Avec W représente la largeur de la carte de forme.

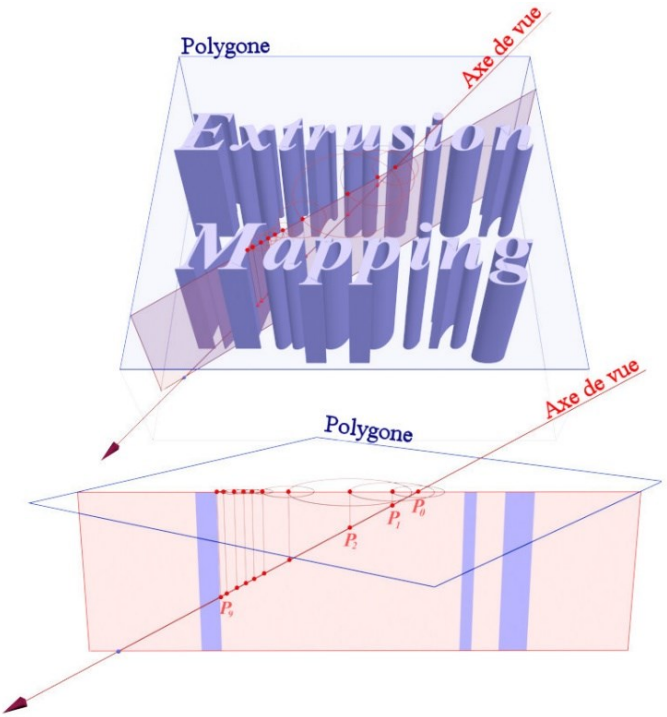


Figure 5.4 : Processus de traçage de rayons associé au placage d'extrusion par pixel [74].

5.2.3 Les types de placage d'extrusion par pixel

En plus de l'extrusion basic, ils existent trois autres types de placage d'extrusion par pixel (Figure 5.5), à savoir, le placage d'extrusion étendue, le placage d'extrusion biseautée et le placage d'extrusion avec chanfrein.

Pour le placage d'extrusion étendue [74] (Figure 5.5b), le terme $EDT(P_i)$ est remplacé par $max(0, EDT(P_i) - e)$, avec e est un paramètre modifiable en temps réel, permettant de moduler l'effet de l'extension, dans ce cas, la distance est calculée par la formule suivante :

$$D(P_i) = \tau \times max(0, EDT(P_i) - e) \tag{5.4}$$

Les autres versions du placage d'extrusion par pixel sont présentées dans [78], à savoir le placage d'extrusion biseautée (Figure 5.5c) qui consiste à créer une extrusion étendue vers l'extérieur qui varie en fonction de la profondeur, et le placage d'extrusion avec chanfrein

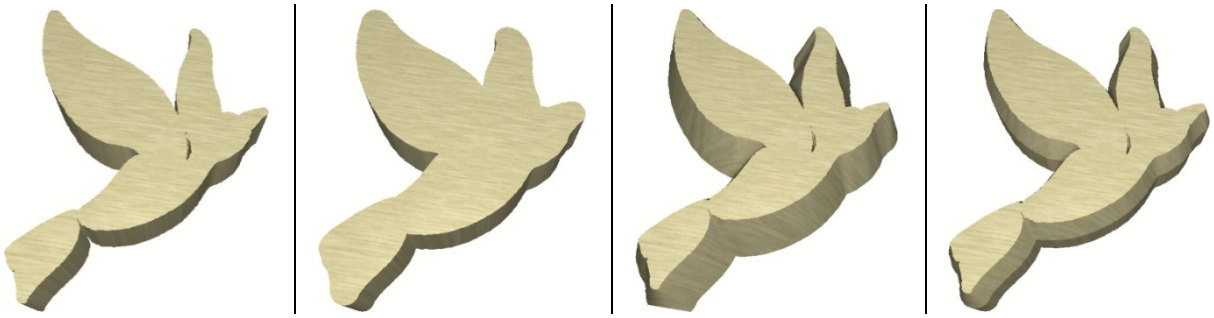
(Figure 5.5d) qui consiste à limiter l'effet du biseau à une certaine profondeur (entre 0 et 1) pour avoir des bords avec un chanfrein.

Concernant le placage d'extrusion biseautée, nous commençons par faire une recherche de l'intersection en se basant sur la distance à la forme d'origine avec la formule (5.2) (Figure 5.6). Pendant cette recherche, il faut tester si la position actuelle est à l'intérieur de la géométrie en utilisant la différence suivante :

$$D_b(P_i) = D(P_i) - bz_i \tag{5.5}$$

Où z_i représente la composante de profondeur de $P_i(x_i, y_i, z_i)$. Et b est un paramètre pour contrôler l'effet du biseau.

Si $D_b(P_i) > 0$ nous continuons la recherche, sinon le point d'intersection trouvé est raffiné avec une recherche binaire en faisant la vérification de la position actuelle si elle est à l'intérieur de la géométrie en utilisant la même distance $D_b(P_i)$.



(a) (b) (c) (d)

Figure 5.5 : (a) Placage d'extrusion basic. (b) Placage d'extrusion étendue. (c) Placage d'extrusion biseautée. (d) Placage d'extrusion avec chanfrein.

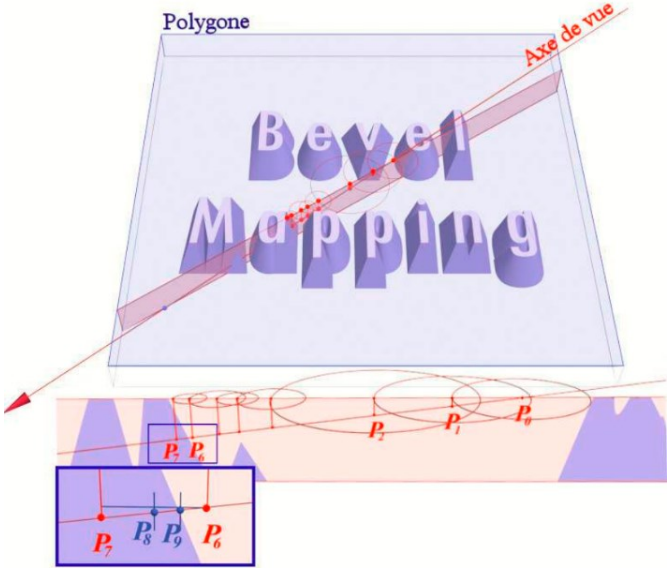


Figure 5.6 : Recherche du point d'intersection dans le cas de l'extrusion biseautée [74].

Concernant le placage d'extrusion avec chanfrein, nous commençons d'abord par effectuer une recherche de l'intersection du rayon de vue avec l'extrusion biseautée (Figure 5.7). Ce point sera l'intersection retenue si sa profondeur est inférieure à une valeur c contrôlant l'effet de chanfrein. Sinon, nous effectuons une deuxième recherche en utilisant la relation suivante :

$$P_{i+1} = P_i + \tau \times max(0, EDT(P_i) - b.c)v = P_i + D_c(P_i)v \tag{5.6}$$

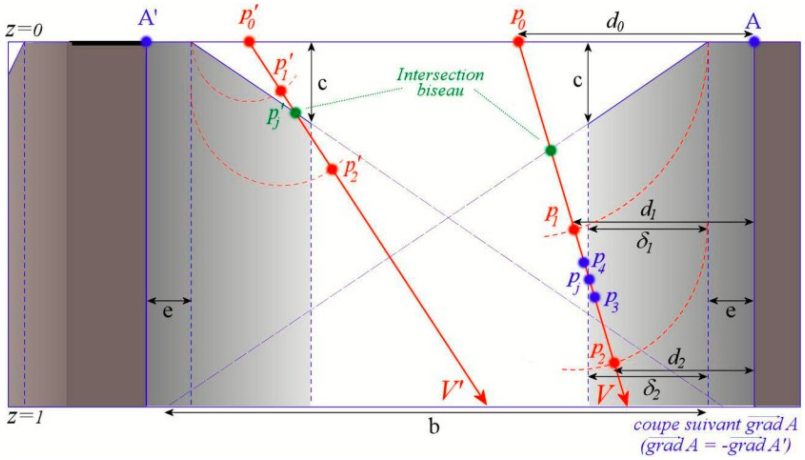


Figure 5.7 : Recherche du point d'intersection dans le cas de l'extrusion avec chanfrein [74]

5.3 PLACAGE D'EXTRUSION PAR PIXEL AVEC CORRECTION DE LA SILHOUETTE

Suivant le même processus utilisé dans le traçage de cônes avec correction de la silhouette, les paramètres de la surface quadratique sont utilisés de la même façon afin que le traçage de cercles prenne en charge les caractéristiques de la surface 3D. Alors nous avons deux approches, la rectification du rayon de vue et la rectification de la distance. La décision concernant les fragments appartenant à la silhouette est effectuée après la phase de traçage de rayons, quel que soit le type de l'extrusion basic, étendu, biseauté ou avec chanfrein. Dans le cas de l'extrusion biseautée, le point trouvé est raffiné avec une recherche binaire (pendant la recherche, il faut tester si la position actuelle est à l'intérieur de la géométrie en utilisant la relation (5.5)) et dans le cas de l'extrusion avec chanfrein nous poursuivons la recherche après la recherche binaire avec la relation (5.6).

5.3.1 Rectification du rayon de vue

La rectification du rayon de vue s'effectue de la même manière que celle effectuée pour le traçage de cônes (Figure 5.8). Les algorithmes Algorithme 5.1 et Algorithme 5.2 implémentent cette rectification dans le cas de l'extrusion basic. Et les algorithmes Algorithme 5.3 et Algorithme 5.4 implémentent cette dernière dans le cas de l'extrusion biseautée et avec chanfrein.

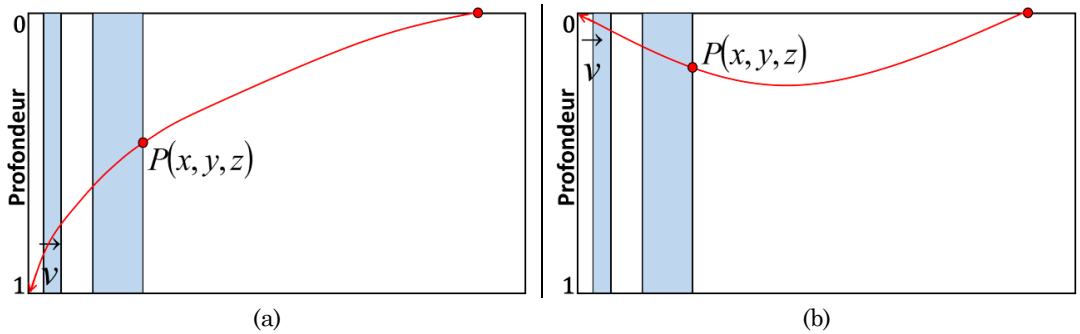


Figure 5.8 : Rectification du rayon de vue. (a) Le rayon de vue est à l'intérieur du quadrique. (b) Le rayon de vue est à l'extérieur du quadrique.

Algorithme 5.1(ExtRectRay1) : Extrusion basique avec rectification du rayon de vue dans le cas où le rayon de vue est à l'intérieur de la quadrique.

```
Entrées : p0, v, w, Cf (Carte de forme)
Sortie : t
Début
 t ← 0
 p ← p0
 Pour i = 1 à nb Faire
  t ← t + D(P)
  p ← p0 + (v + w)*t
 FinPour
Fin
```

Algorithme 5.2 (ExtRectRay2) : Extrusion basique avec rectification du rayon de vue dans le cas où le rayon de vue est à l'extérieur de la quadrique.

```
Entrées : p0, v, d, Cf (Carte de forme)
Sortie : t
Début
 t ← 0
 p ← p0
 Pour i = 1 à nb Faire
  t ← t + D(P)
  p ← p0 + v*t
  p.z ← p.z - t*t*d
 FinPour
Fin
```

Algorithme 5.3 (ExtBiseauRectRay1) : Extrusion biseautée avec rectification du rayon de vue dans le cas où le rayon de vue est à l'intérieur de la quadrique.

```
Entrées : p0, v, w, Cf (Carte de forme)
Sortie : t, dist
Début
 t ← 0
 p ← p0
 Pour i = 1 à nb Faire
  Si (Db(P)> 0) Alors
   dist ← D(P)
   t ← t + D(P)
   p ← p0 + (v + w)*t
  FinSi
 FinPour
Fin
```

Algorithme 5.4 (ExtBiseauRectRay2) : Extrusion biseautée avec rectification du rayon de vue dans le cas où le rayon de vue est à l'extérieur de la quadrique.

```
Entrées : p0, v, d, Cf (Carte de forme)
Sortie : t, dist
Début
 t ← 0
 p = p0
 Pour i = 1 à nb Faire
  Si (Db(P)> 0) Alors
   dist ← D(P)
   t ← t + D(P)
   p ← p0 + t*v
   p.z ← p.z - t*t*d
  FinSi
 FinPour
Fin
```

5.3.2 Rectification de la distance

La rectification de la distance consiste à adapter le traçage de cercles (Figure 5.9) pour qu'il prenne en charge les deux formes de la surface quadratique. Dans le cas de l'extrusion, nous avons deux valeurs de profondeur au niveau de la carte de forme qui sont 0 et 1. Alors la rectification consiste à modifier la valeur de la distance $D(P_i)$ selon les paramètres de la surface quadratique.

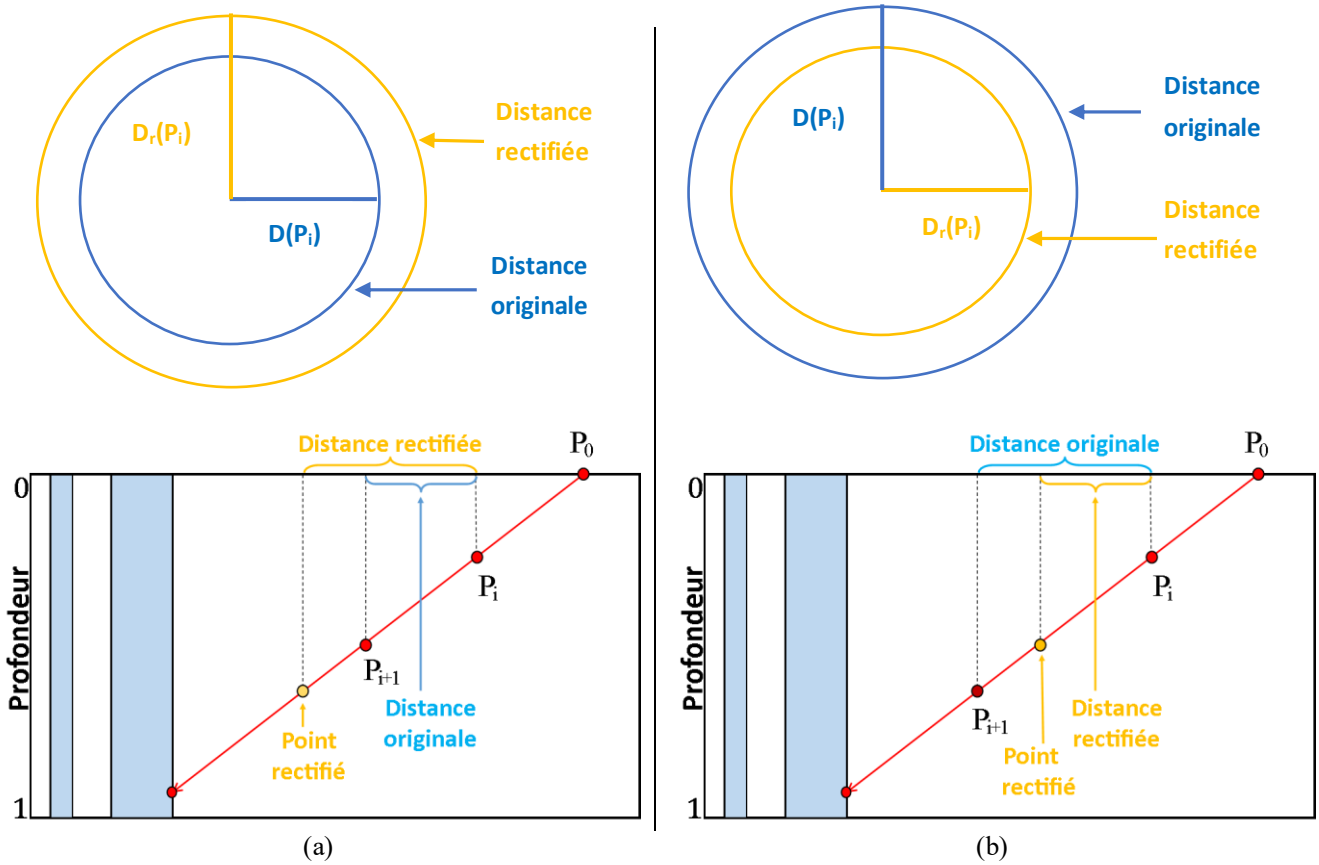


Figure 5.9 : Rectification de la distance. (a) La distance est augmentée afin que la surface quadratique se rapproche du point P_{i+1}. (b) La distance est diminuée afin que la surface quadratique s'éloigne du point P_{i+1}.

Dans le premier cas (Figure 4.13a), la distance $D(P)$ doit s'agrandir pour qu'on s'approche de la surface quadratique, alors la rectification consiste à augmenter la valeur de l'$EDT(P)$ en utilisant la distance d et le vecteur u selon la formule :

$$D_r(P) = \tau \times \left(EDT(P) + d\|u_{xy}\|t\right) \tag{5.7}$$

La Figure 5.9a montre que le déplacement avec la distance rectifiée est plus rapide que celui de la distance originale, car le pas de déplacement augmente. A chaque itération, la rectification de la distance fait avancer le point P_{i+1} le long du rayon de vue, et dans le cas d'une intersection, on converge rapidement.

Dans le deuxième cas (Figure 4.13b), la distance $D(P)$ diminue pour qu'elle s'éloigne de la surface quadratique, alors la rectification consiste à diminuer la valeur de l'$EDT(P)$ en utilisant simplement la distance d selon la formule :

$$D_r(P) = \tau \times EDT(P) \times (1 - t^2 d) \tag{5.8}$$

La Figure 5.9b montre que le déplacement avec la distance rectifiée est plus lent que celui de la distance originale, car le pas de déplacement diminue. A chaque itération, la

rectification de la distance fait reculer le point P_{i+1} le long du rayon de vue, et dans le cas où il n'y a pas d'intersection, on diverge.

Dans les deux cas, la valeur t_{i+1} est calculée par :

$$t_{i+1} = t_i + D_r(P_i) \tag{5.9}$$

Et le prochain point P_{i+1} est calculé avec la relation (4.2). A la fin de de la recherche de l'intersection, on vérifie si $t > t_{max}$, si tel est le cas, le pixel est ignoré.

Concernant le placage d'extrusion biseautée et avec chanfrein, la distance $D_b(P_i)$ doit être aussi rectifiée selon la formule suivante :

$$D_{rb}(P_i) = D_r(P_i) - bz_i \tag{5.10}$$

Si $D_{rb}(P_i) > 0$ nous continuons la recherche, sinon le point d'intersection trouvé est raffiné par une recherche binaire.

L'Algorithme 5.5 implémente cette rectification dans le cas de l'extrusion basic. Et l'Algorithme 5.6 implémente cette dernière dans le cas de l'extrusion biseautée et avec chanfrein.

Algorithme 5.5 (ExtRectDist) : Extrusion basique avec rectification de la distance.

```
Entrée : p0, v, Cf (Carte de forme)
Sortie : t
Début
 t ← 0
 p ← p0
 Pour i = 1 à nb Faire
  t ← t + Dr(P)
  p ← p0 + v*t
 FinPour
Fin
```

Algorithme 5.6 (ExtBiseauRectDist) : Extrusion biseautée avec rectification de la distance.

```
Entrée : p0, v, Cf (Carte de forme)
Sortie : t, dist
Début
 t ← 0
 p ← p0
 Pour i = 1 à nb Faire
  Si (Drb(P)> 0) Alors
   dist ← Dr(P)
   t ← t + Dr(P)
   p ← p0 + v*t
  FinSi
 FinPour
Fin
```

5.3.3 Algorithme de placage d'extrusion avec correction de la silhouette

L'algorithme Algorithme 5.7 implémente le placage d'extrusion avec correction de la silhouette en prenant en compte ses différents types. L'organigramme de la Figure 5.10 résume les étapes principales de l'algorithme. En plus des notations adopter pour l'Algorithme 4.7, on note de plus :

- C_f : Carte de forme.
- τ : Facteur d'échelle de la distance.
- e, b, c: Paramètres de l'extension, de biseau et de chanfrein.
- $siBiseau$: s'il y a un placage biseauté.
- $siChanfrein$: s'il y a un placage avec chanfrein.

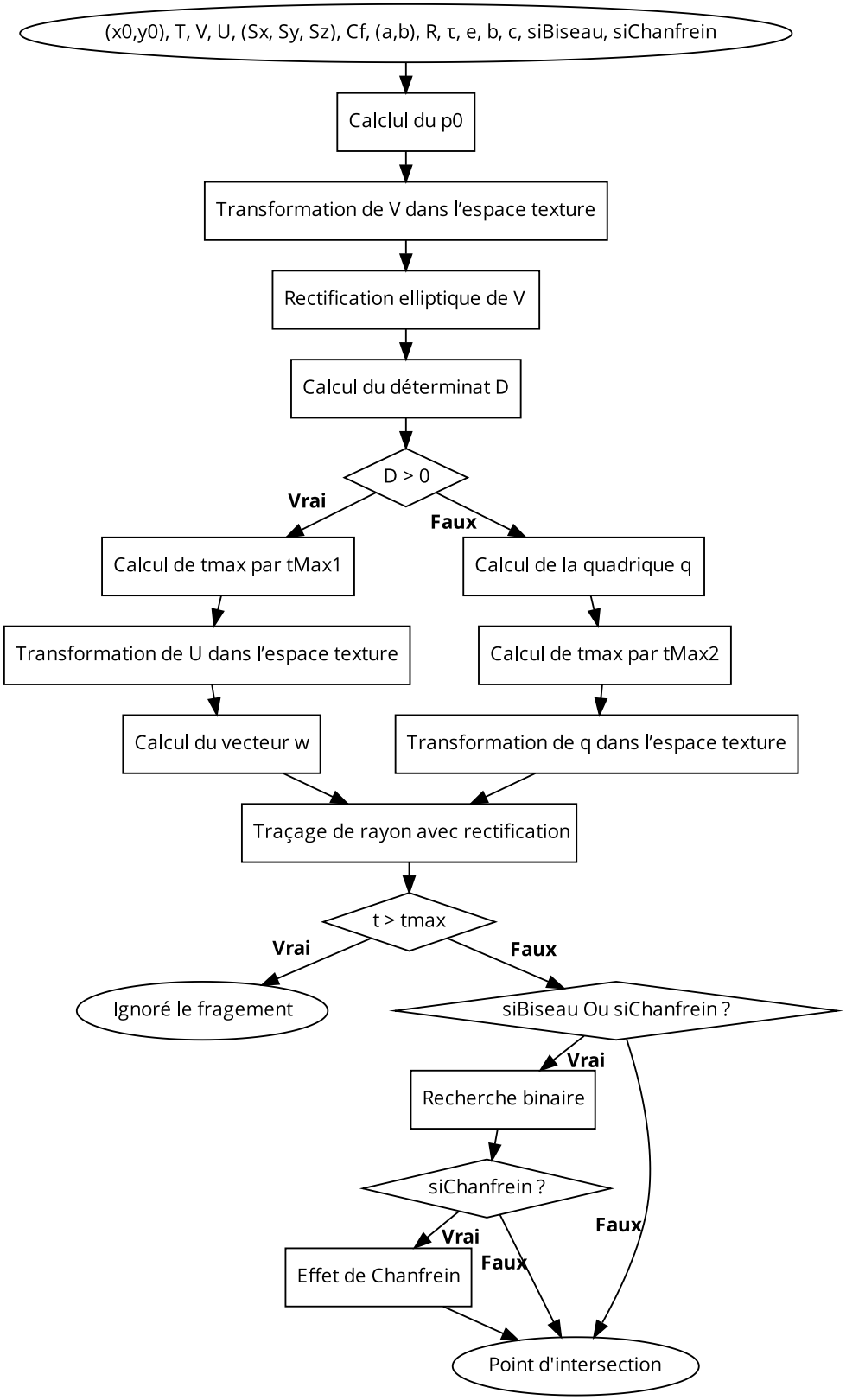


Figure 5.10 : Organigramme de l'algorithme de placage d'extrusion avec pris en charge de la silhouette.

Algorithme 5.7 : Placage d'extrusion par pixel avec pris en charge de la silhouette.

```
Entrée : (x0,y0), T, V, U, (Sx, Sy, Sz), Cf, (a,b), R, τ, e, b, c, siBiseau, siChanfrein
Sortie : p
Début
 p0 ← (T.x*x0, T.y*y0, 0)
 v ← Normaliser(V/(Sx, Sy, Sz))          //Transformation dans l'espace texture
 v.z ← - v.z
 z ← v.z
 v ← v/v.z
 //Rectification elliptique pour les textures non carrées
 vR ←(Norme(v.xy)/(Racine(v.x*v.x+R*R*v.y*v.y)
 A ← a*U.x*U.x + b*U.y*U.y
 B ← 2*a*V.x*U.x + 2*b*V.y*U.y - U.z
 C ← a*V.x*V.x + b*V.y*V.y - V.z
 D ← B*B - 4*A*C
 Si D > 0 Alors
  tmax ← tMax1(V, U, a, b)
  u ← Normalizer(U/(Sx, Sy, Sz))          //Transformation dans l'espace texture
  w ← ((B - Racine(D))/-2*A)*u/z
   Si siBiseau Ou siChanfrein Alors
    t ← ExtBiseauRectRay1(p0, vR, w, Cf)   //Rectification du rayon de vue
    // t ← ExtBiseauRectDist1(p0, vR, Cf)  //Rectification de la distance
   Sinon
    t ← ExtRectRay1(p0, vR, w, Cf)         //Rectification du rayon de vue
    // t ← ExtRectDist1(p0, vR, Cf)        //Rectification de la distance
   FinSi
 Sinon
  q ← a*V.x*V.x + b*V.y*V.y;
  tmax ← tMax2(V, q)
  q ← (q/Sz)/(vz*vz)                       //Transformation dans l'espace texture
  Si siBiseau Ou siChanfrein Alors
   t ← ExtBiseauRectRay2(p0, vR, q, Cf)    //Rectification du rayon de vue
   // t ← ExtBiseauRectDist2(p0, vR, Cf)   //Rectification de la distance
  Sinon
   t ← ExtRectRay2(p0, vR, q, Cf)          //Rectification du rayon de vue
   // t ← ExtRectDist2(p0, vR, Cf)         //Rectification de la distance
  FinSi
 FinSi
 Si t > tmax Alors
  Ignoré
 Sinon
  p = p0 + vR*t
 FinSi
 Si siBiseau Ou siChanfrein Alors
  Pour i = 1 à nb Faire
   dist ← dist/2
   Si (Db(P)> 0) Alors
    P ← P + dist * v
   Sinon
    P ← P - dist * v
   FinSi
  FinPour
 FinSi
 Si siChanfrein Alors
  Si p.z > c Alors
   Pour i = 1 à nb Faire
    P ← P + Dc(P)*v
   FinPour
  FinSi
 FinSi
Fin
```

5.4 RESULTAT ET DISCUSSION

Les images des figures sont rendues avec 25 étapes linéaires et 5 étapes binaires en exploitant les deux configurations matérielles du Tableau 0.2. Pour la résolution des textures, nous avons utilisé dans la plupart des cas la résolution 512×512. D'après notre expérimentation, la qualité de rendu des deux approches est presque identique, alors nous avons choisi de ne pas mentionner le nom de l'approche utilisée dans les images des figures.

L'objectif principal du placage d'extrusion avec correction de la silhouette est d'éliminer les parties non visibles de la silhouette. En effet, les figures Figure 5.11 et Figure 5.12 montrent la différence entre les objets rendus avec la technique de placage d'extrusion sans et avec correction de la silhouette et en utilisant différentes échelles de profondeur. On remarque que les différences de rendu sont visibles sur les bords des objets 3D.

Les deux rectifications proposées fonctionnent aussi bien dans le cas de l'extrusion biseautée et avec chanfrein, la Figure 5.13 montre que les effets de biseau et de chanfrein ne sont pas affectés par cette correction, seuls les fragments appartenant à la silhouette sont concernés. La même chose s'applique également dans le cas de l'extrusion étendue, la Figure 5.14 montre l'effet de l'extension sur des objets 3D en augmentant progressivement la valeur de l'extension e. L'extrusion étendue peut être combinée avec toute autre technique de placage d'extrusion sans affecter le processus de traçage de rayons. La Figure 5.5 montre que dans les trois autres types d'extrusion (extrusion de base, biseautée et avec chanfrein), nous avons ajouté l'effet de l'extension. Il en est de même pour la correction de la silhouette.

La rectification proposée ne dépend pas de la carte de forme, elle dépend uniquement des paramètres quadratiques associés à la surface 3D. De plus, la carte de forme n'est pas attachée à la géométrie de base sur laquelle elle est appliquée, car le processus de rectification est réalisé en temps réel. Cela permet d'utiliser le même processus de rectification et la même texture en temps réel sur différents objets 3D. Alors le processus de rectification et la carte de forme sont indépendants de la surface sur laquelle ils seront utilisés. De plus, le changement de la position de la caméra n'influence pas le processus de rectification ni la qualité de rendu, puisque la rectification est réalisée en temps réel et à chaque mouvement de la caméra, nous aurons une nouvelle image rendue avec une nouvelle rectification.

Pour comparer la vitesse de rendu, nous avons utilisée trois cartes de forme différentes qui ont une résolution de 1024×1024 et avec des répétitions selon x et y. Les tableaux Tableau 5.1 et Tableau 5.2 montrent des comparaisons de la vitesse de rendu entre les techniques de placage d'extrusion et la technique du placage de relief en prenant en compte le traitement de la silhouette. Les figures Figure 5.15 et Figure 5.16 montre des comparaisons de la vitesse moyenne de rendu.

Le Tableau 5.1 et la Figure 5.15 montrent la différence de la vitesse de rendu de l'approche par rectification du rayon de vue par rapport aux techniques de placage d'extrusion [74] et le placage de relief [84] en utilisant la configuration matériel *Config.1* (Tableau 0.2).

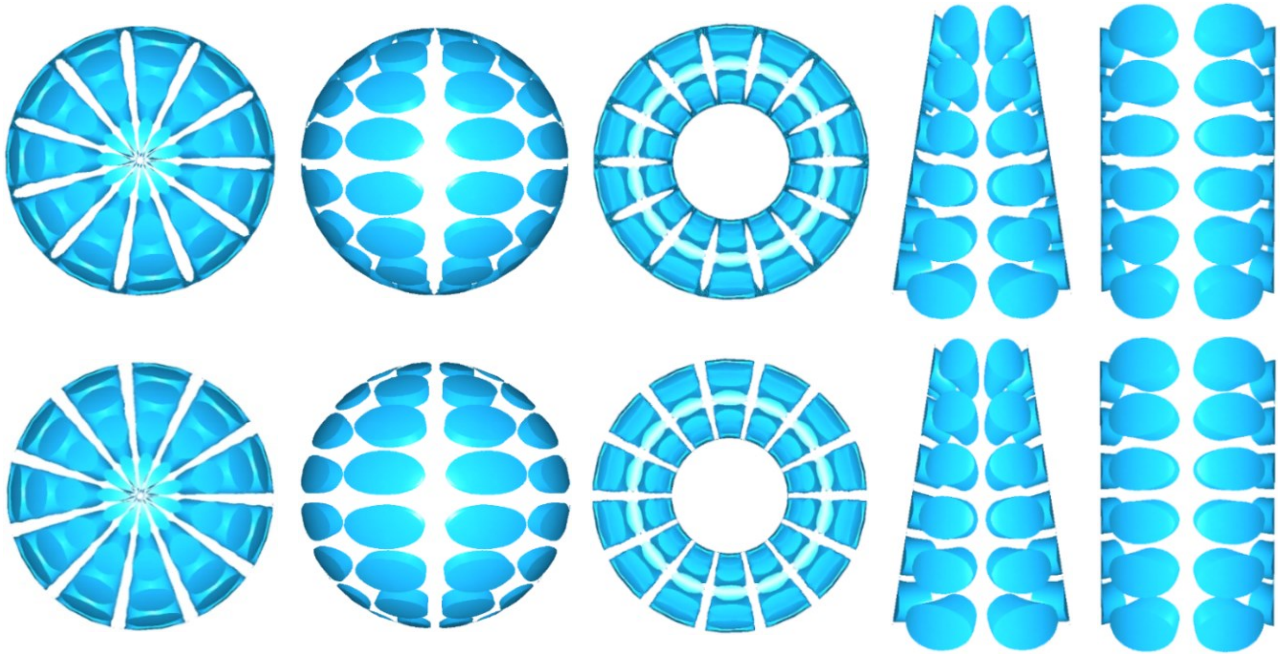


Figure 5.11 : Comparaison de rendu sans et avec correction de silhouette. En haut, le placage d'extrusion basique et en bas le placage d'extrusion avec correction de la silhouette.

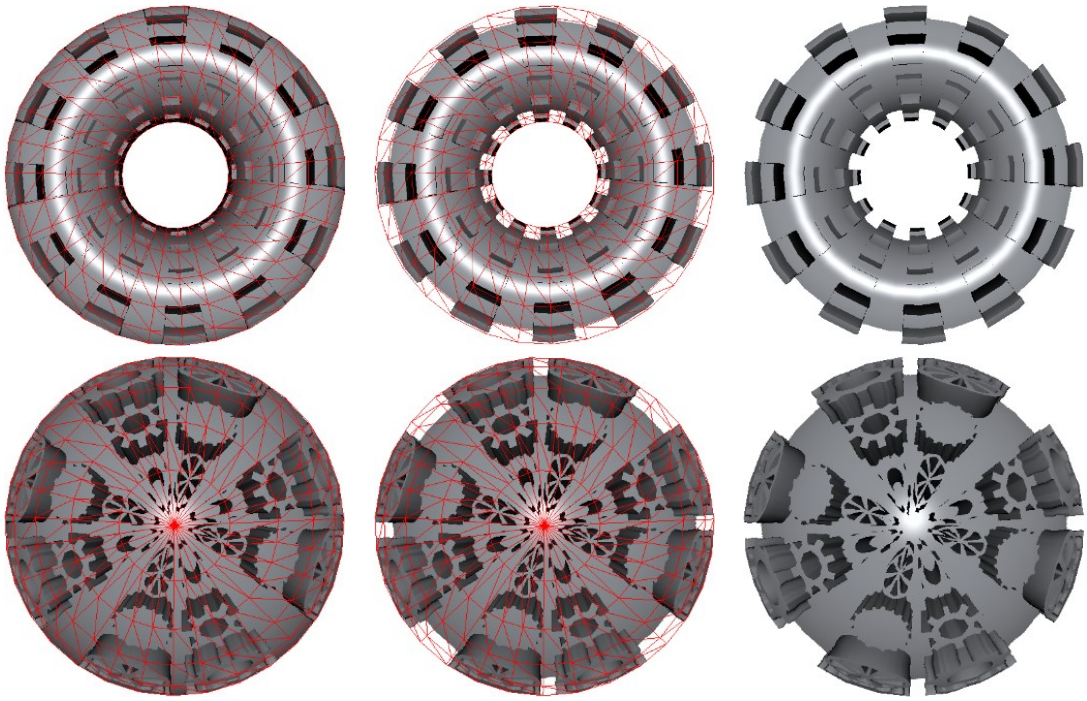


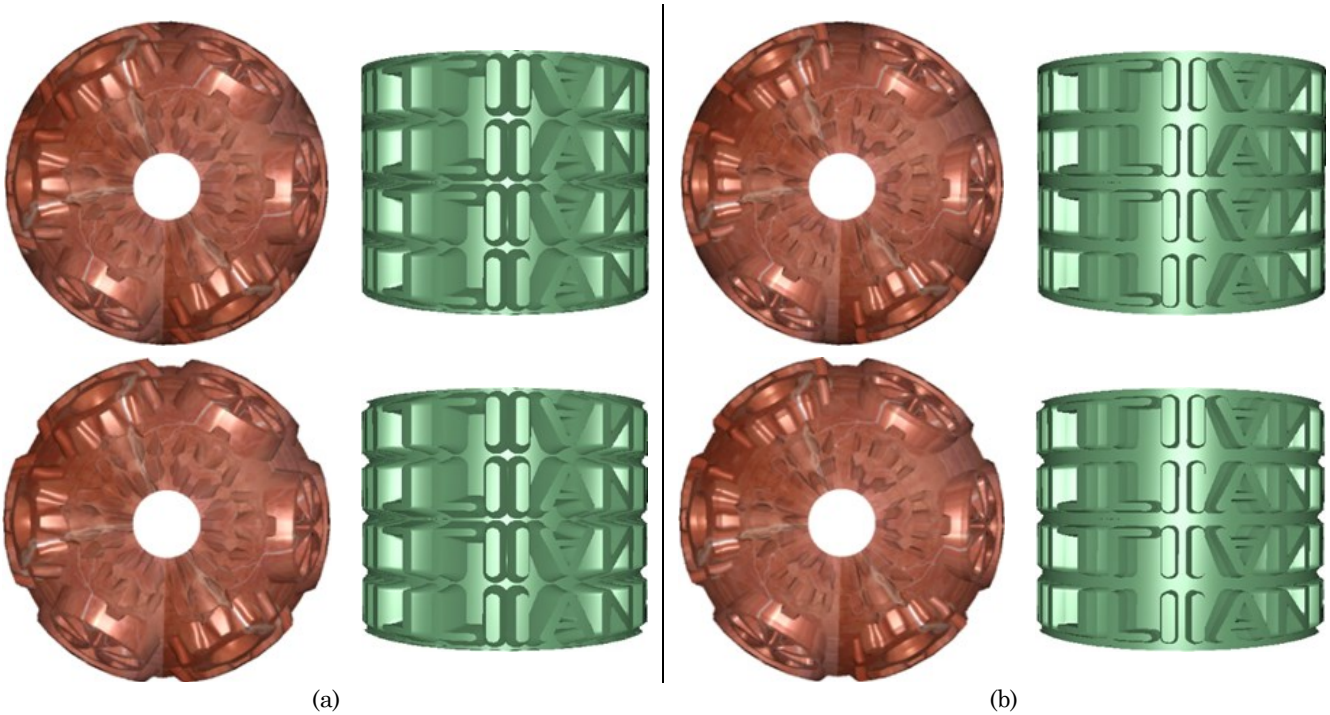
Figure 5.12 : Comparaison de rendu du placage d'extrusion sans et avec correction de la silhouette.

Nous observons que les différentes techniques de placage d'extrusion sont plus rapides par rapport à la technique de placage de relief. Dans le cas du placage d'extrusion avec rectification du rayon de vue, nous avons une différence moyenne considérable de 42IPS, 26IPS et 18IPS respectivement pour l'extrusion basique, biseautée et avec chanfrein. Nous notons également que la vitesse de rendu des techniques de placage d'extrusion avec correction de la silhouette a été réduite par rapport aux techniques de placage d'extrusion par un moyen de 8IPS. La dégradation est minime et elle est due aux calculs liés au traitement de la silhouette basé sur l'approximation quadratique.

Le Tableau 5.2 et la Figure 5.16 montrent la différence de la vitesse de rendu entre la rectification du rayon de vue et la rectification de la distance en utilisant la configuration

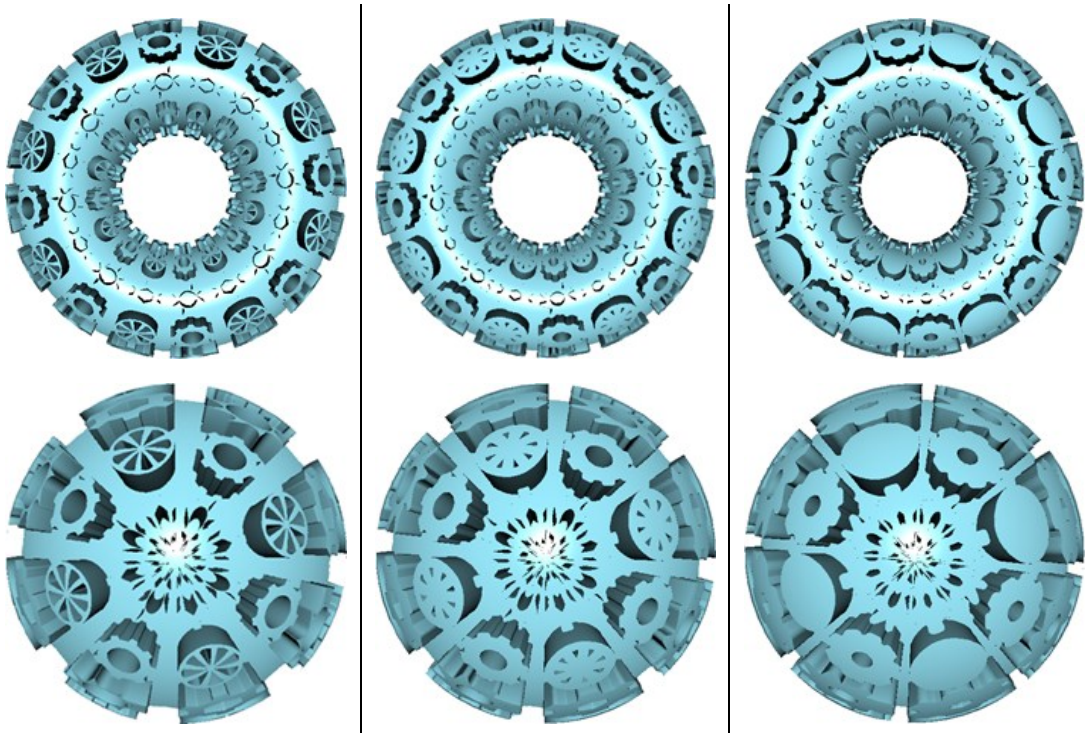
matériel *Config.2* (Tableau 0.2). Dans le cas de l'extrusion basique, nous observant que l'approche par rectification de la distance est la plus rapide à une différence de 8IPS. Quant aux autres types d'extrusion , la vitesse est la même dans les deux rectifications.

La combinaison des techniques du placage d'extrusion par pixel et l'approximation quadratique produit des résultats satisfaisants à faible coût. Cependant, elle souffre du même problème que celui évoqué dans le chapitre précédant (Figure 5.17).



(a) (b)

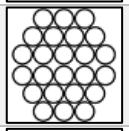
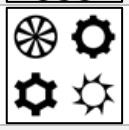
Figure 5.13 : Placage d'extrusion biseautée (a) et avec chanfrein (b) sans et avec correction de la silhouette.



$e = 0$ $e = 0.4$ $e = 0.8$

Figure 5.14 : Placage d'extrusion étendue avec correction de la silhouette.

Tableau 5.1 : Comparaison de la vitesse de rendu IPS (Images Par Seconde) de l'approche par rectification du rayon de vue avec le placage d'extrusion [74] et le placage de relief [84].

Scène 800×600 		Placage d'extrusion [74]			Placage d'extrusion avec rectification du rayon de vue			Placage de relief avec correction de la silhouette [84]
Carte de forme	(T_x,T_y)	Basique	Biseautée	Avec chanfrein	Basique	Biseautée	Avec chanfrein	
	(6,6)	200	184	176	192	176	168	136
	(10,10)	160	144	136	152	136	128	128
	(12,10)	152	136	128	144	128	120	96
Moyenne		**170**	**154**	**146**	**162**	**146**	**138**	120

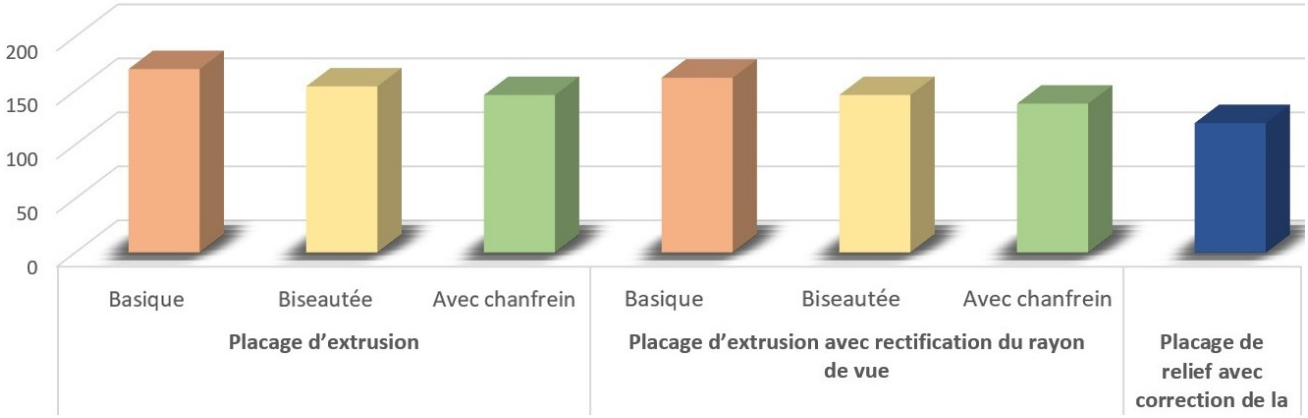
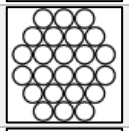
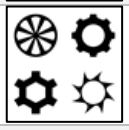


Figure 5.15 : Comparaison de la vitesse moyenne de rendu de l'approche par rectification du rayon de vue avec le placage d'extrusion [74] et le placage de relief [84].

Tableau 5.2 : Comparaison de la vitesse de rendu IPS (Images Par Seconde) entre les deux approches du placage d'extrusion avec correction de la silhouette.

Scène 800×600 		Rectification du rayon de vue			Rectification de la distance		
Carte de forme	(T_x,T_y)	Extrusion Basique	Extrusion biseautée	Extrusion avec chanfrein	Extrusion Basique	Extrusion biseautée	Extrusion avec chanfrein
	(6,6)	232	160	148	240	160	148
	(10,10)	165	108	98	172	108	99
	(12,10)	164	101	92	171	100	92
Moyenne		**187**	**123**	**113**	**195**	**123**	**113**

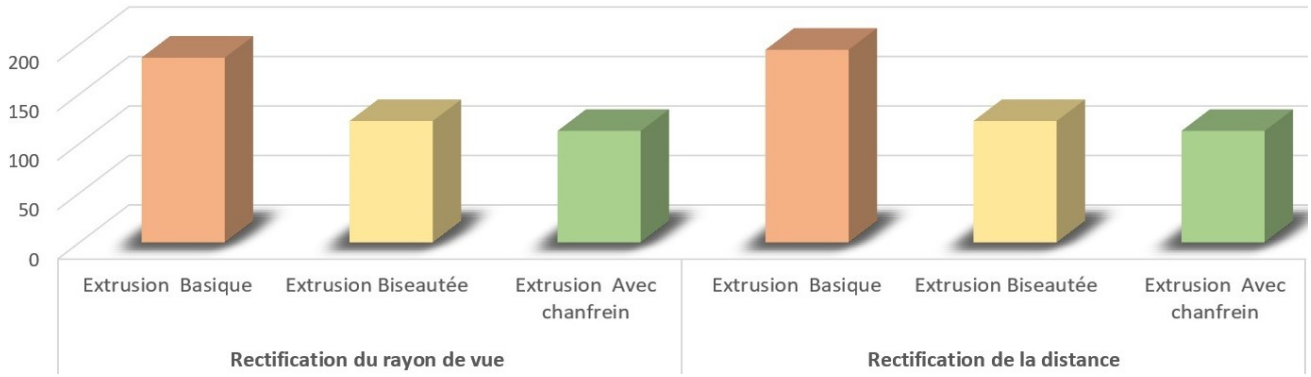


Figure 5.16 : Comparaison de la vitesse moyenne de rendu des deux rectifications du placage d'extrusion.

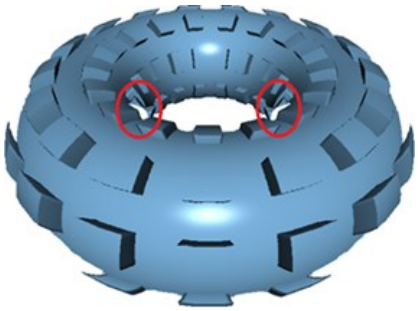


Figure 5.17 : Distorsions dus à l'utilisation de l'approximation quadratique.

5.5 CONCLUSION

Dans ce chapitre, nous avons présenté une amélioration concernant le placage d'extrusion par pixel qui consiste à exploiter les paramètres de la surface quadratique afin de déterminer les fragments appartenant à la silhouette. Alors nous avons proposée deux nouveaux algorithmes de traçage de rayons pour chacune des méthodes d'extrusion basique, étendue, biseautée et avec chanfrein.

Durant la phase de rendu, le premier algorithme consiste à utiliser les paramètres de la surface quadratique pour rectifier le rayon de vue et le deuxième algorithme consiste à utiliser ces paramètres pour rectifier les distances extraites de la carte de forme. En général, les qualités des images rendues par les deux approches sont identiques au niveau des différentes techniques de placage d'extrusion. Cependant, la comparaison de la vitesse de rendu a révélé que l'approche par rectification de la distance est la plus rapide dans le cas de l'extrusion basique. Mais dans le cas de l'extrusion biseautée et avec chanfrein, la vitesse de rendu set presque la même.

Conclusion et Perspectives

BILAN

Le rendu en temps réel joue un rôle crucial dans le domaine de l'infographie 3D, notamment en ce qui concerne la synthèse d'images. Assurer une interaction fluide au sein d'une scène 3D, constitue une priorité essentielle pour garantir une expérience de qualité pour les utilisateurs. Cependant, il est important de noter que plus une scène 3D se rapproche du réalisme, plus le temps de calcul nécessaire pour générer le rendu augmente de manière significative. Cela devient particulièrement préoccupant lorsque la scène est composée d'une variété d'objets 3D complexes.

Les approches informatiques de modélisation basées sur des maillages polygonaux rencontrent encore certaines limitations. Lorsqu'il est nécessaire de simplifier un maillage pour des raisons de performances, cela peut entraîner des problèmes visuels tels que des artefacts ou une perte de détails. De plus, l'optimisation de ces processus grâce à des algorithmes appropriés est essentielle, car le rendu des scènes contenant un très grand nombre de primitives peut entraîner une diminution significative des performances de rendu, compromettant ainsi l'interactivité en temps réel.

D'autre part, la représentation des mésostructures pose un défi majeur dans le domaine du rendu 3D, car elles ne peuvent être entièrement représentées ni par un maillage polygonal, en raison de contraintes de performances, ni par des techniques d'ombrage classiques, en raison de leur pauvreté visuelle. Les mésostructures font référence aux structures à une échelle intermédiaire entre les détails géométriques microscopiques et les formes globales macroscopiques d'un objet 3D.

Nos travaux de recherche sont concentrés sur le développement de solutions innovantes, visant à produire une qualité de rendu appréciable, tout en conservant l'interactivité des applications 3D en temps réel. Ces solutions peuvent impliquer l'utilisation de techniques avancées de modélisation et de rendu, telles que la représentation procédurale, la gestion efficace des ressources et l'optimisation des algorithmes de rendu. L'objectif ultime est d'atteindre un équilibre entre la complexité des scènes 3D, la qualité visuelle et les performances de rendu, afin d'offrir une expérience interactive et réaliste aux utilisateurs.

Tout au long de cette thèse, nous avons exposé en détail nos approches qui ont été axées sur l'amélioration des techniques de placage de déplacement tridimensionnel par pixel, notamment, *la technique du placage de relief*, *la technique de traçage de cônes* et *la technique de placage d'extrusion*. Notre objectif principal était de créer des modèles 3D avec un niveau élevé de réalisme tout en offrant des performances optimales. Nos approches ont permis de maximiser le réalisme des objets rendus, tout en garantissant une expérience utilisateur interactive. Nous avons réussi à exploiter les meilleures pratiques et techniques disponibles afin d'optimiser le processus de rendu, d'améliorer la qualité des modèles 3D et de créer des environnements attractifs. Alors nous avons proposé quatre contributions majeures.

Nous avons commencé par proposer une synchronisation en temps réel, combinée entre l'échelle de profondeur des microreliefs, l'angle de vue et le nombre d'itérations de l'algorithme de traçage de rayons avec l'ombrage correspondant. Cette synchronisation est

bien utile pour les techniques de placage de déplacement tridimensionnel par pixel, notamment, la technique de placage de relief dont la qualité de rendu dépend de ces trois facteurs. En effet, le placage de relief produit une qualité de rendu acceptable à un coup relativement faible, mais seulement dans le cas où l'échelle de profondeur des microreliefs est minimale, dans le cas contraire, ou si l'angle de vue rase la surface de l'objet 3D, celui-ci est rendu avec des problèmes d'artefact. Alors la synchronisation proposée est une solution flexible et dynamique qui s'adapte aux variations de l'échelle de profondeur et de l'angle de vue, offrant ainsi des rendus corrects dans différentes conditions.

Ensuite, nous avons introduit une troisième version de la technique de traçage de cônes, qu'on l'a appelée la technique hybride. La technique de traçage de cônes utilise une phase de prétraitement pour calculer l'espace vide autour de chaque pixel de la carte de profondeurs, sous forme de cônes ouverts vers le haut. La technique a été présentée en deux versions, la technique conservative et la technique relaxée, et chacune de ces techniques possèdes des avantages et des inconvenants, offrant ainsi des choix adaptés aux caractéristiques des objets 3D et aux spécificités du projet. Alors notre proposition est de créer un interpolateur entre le cône conservatif et le cône relaxé, afin de remédier aux problèmes lies aux deux techniques.

Malgré l'efficacité de la technique de traçage de cônes, elle reste bien adaptée uniquement pour les surfaces planes. En effet, quand il s'agit des surfaces courbées, le traitement de la silhouette n'est pas pris en charge, ce qui rend le rendu des objets 3D incorrecte. Pour surmonter cette limitation, nous avons proposé deux nouveaux algorithmes de traçage de cônes, qui exploitent une approximation quadratique de la surface de l'objet 3D au niveau de chaque sommet du maillage polygonal correspondant. Ces algorithmes permettent de rendre correctement les microreliefs, et de déterminer les parties de l'objet appartenant à la silhouette. Bien que les deux algorithmes proposés offrent une qualité visuelle similaire, des différences significatives ont été observées concernant la vitesse de rendu.

A l'instar de la technique de traçage de cônes, nous avons introduit une amélioration pour le placage d'extrusion par pixel en se basant aussi sur l'approximation quadratique afin de déterminer les parties appartenant à la silhouette. Alors nous avons proposé deux nouveaux algorithmes de traçage de rayons pour chacune des méthodes d'extrusion basique, étendue, biseautée et avec chanfrein. Le placage d'extrusion permet de créer des motifs extrudés sur des objets 3D, en exploitant les principes du placage de déplacement tridimensionnel par pixel. Globalement, la qualité des images obtenues par les deux algorithmes est similaire pour les différentes techniques d'extrusion. Cependant, des disparités de vitesse de rendu ont été observées dans le cas de l'extrusion basique.

Les solutions proposées dans cette thèse offrent des améliorations significatives aux meilleures techniques de placage de déplacement tridimensionnel par pixel. La synchronisation en temps réel permet d'ajuster dynamiquement l'échelle de profondeur, l'angle de vue et le nombre d'itérations de l'algorithme de traçage de rayons, garantissant ainsi des rendus précis et de haute qualité dans différentes conditions. Les techniques de traçage de cônes hybrides et de traçage de cônes avec correction de la silhouette améliorent respectivement le rendu des surfaces planes et courbées, permettant une représentation plus

réaliste des objets 3D. De plus, les approches de traçage de rayons pour le placage d'extrusion offrent des rendus visuellement améliorés et efficaces, adaptés à différentes méthodes d'extrusion. Ces avancées contribuent à l'amélioration globale du rendu 3D en temps réel dans le domaine de l'infographie 3D, ouvrant de nouvelles possibilités pour des applications telles que les jeux vidéo, les simulations interactives et la réalité virtuelle ou augmentée.

PERSPECTIVES

Les études réalisées et les contributions proposées dans cette thèse ouvrent de nombreuses perspectives prometteuses. Tout d'abord, elles permettent d'améliorer la qualité et le réalisme du rendu 3D, tout en conservant une interactivité fluide au sein des scènes 3D. Ces contributions offrent la possibilité de créer des modèles 3D réalistes, tout en maintenant des performances optimales, ce qui est essentiel pour les applications interactives en temps réel. Par exemple, dans le contexte du rendu de scènes complexes avec de nombreux objets 3D, les nouvelles approches permettent de surmonter les problèmes de perte de détails ou d'artefacts visuels.

Les perspectives envisagées pour faire évoluer nos travaux de recherche sont assez nombreuses. Celles qui nous apparaissent les plus importantes et qu'elles constituent, par conséquent, la suite logique de cette thèse, sont résumées dans les paragraphes suivants :

Concernant la technique de traçage de cônes hybrides, le calcul du nouveau cône est réalisé durant la phase de rendu. A chaque itération, on récupère le cône conservatif et le cône relaxé à partir de deux cartes de déplacement, et on calcule le cône hybride. Malgré que cette technique améliore la qualité de rendu, néanmoins, le double accès à la carte de déplacement diminue la vitesse de l'algorithmes de traçage de cônes. Alors il serait plus judicieux de calculer les cônes hybrides dans une phase de prétraitement, soit à partir des deux cônes conservatifs et relaxés ou en proposant une nouvelle méthode.

La combinaison des deux techniques de traçage de cônes et du placage d'extrusion par pixel avec l'approximation quadratique a considérablement résolu le problème de la silhouette aux bords des objets 3D, et a permis d'augmenter leur réalisme. Mais, comme il a été mentionné dans les deux chapitres précédents, dans certains cas, des distorsions peuvent être apparus. Alors plusieurs solutions sont envisageables, vue que la surface approximative est calculée par sommet, et si on part du fait que son interpolation durant la phase de rendu n'est pas fiable, la solution est de stocker les paramètres de cette surface dans une texture 2D, est de l'exploiter durant la phase de rendu, cette solution permet d'avoir une cohérence entre l'espace objet et l'espace texture pour l'algorithme de traçage de rayons. La deuxième solution est de considérer l'espace tangent au lieu de l'approximation quadratique pour la représentation locale de l'objet au niveau de chaque sommet. L'avantage de cette solution est que l'espace tangent est toujours inclus aux maillages polygonaux et ne nécessite aucune étape de prétraitement. La troisième solution est d'utiliser le placage à l'aide de coquille, qui repose sur la création d'une coquille au niveau de chaque triangle du maillage polygonal en utilisant le principe de l'extrusion de sommets. Cette dernière solution s'avère plus couteuse,

vue qu'elle ajoute des primitives supplémentaires au maillage de base, ce qui pourrait diminuer la vitesse de rendu.

En exploitant les améliorations proposées dans cette thèse, les techniques de placage de déplacement tridimensionnel par pixel pourraient être une alternative prometteuse aux techniques de placage de déplacement à base de maillages polygonaux. Pour prendre en compte l'éclairage dans les scènes 3D, la plupart des techniques sont combinées avec des modèles d'ombrage. Or, cette combinaison permet d'avoir seulement une illumination locale qui est propre à chaque objet 3D, et ne considère aucun autre aspect environnemental de la scène. Alors des perspectives futures pourraient inclure l'exploration de nouvelles solutions qui prennent en compte d'autres aspects de rendu, tels que l'illumination globale et le placage d'environnement, pour obtenir des résultats encore plus réalistes. De plus, l'utilisation des techniques d'accélération matérielle ou la parallélisation des calculs pourrait permettre d'optimiser davantage les performances de rendu.

PUBLICATIONS

Journaux internationaux

Adnane Ouazzani Chahdi, Anouar Ragragui, Akram Halli and Khalid Satori, **Per-Pixel Displacement Mapping Using Cone Tracing with Correct Silhouette**, *Journal of Graphic Engineering and Design*, Volume 12 (4), pp. 39–61 2021. DOI : 10.24867/JGED-2021-4-039.

Adnane Ouazzani Chahdi, Anouar Ragragui, Akram Halli and Khalid Satori, **Per-Pixel Extrusion Mapping with Correct Silhouette**, *Computer Science* 22(3), pp. 403–428, 2021. DOI : 0.7494/csci.2021.22.3.3337.

Conférences internationales

Adnane Ouazzani Chahdi, Anouar Ragragui, Akram Halli and Khalid Satori, **Dynamic relief mapping**, *International Conference on Intelligent Systems and Computer Vision (ISCV)*, *IEEE Morocco section*, ISBN 978-1-5386-4396-9, April 2018. DOI: 10.1109/ISACV.2018.8354053.

Adnane Ouazzani Chahdi, Anouar Ragragui, Akram Halli and Khalid Satori, **Per-Pixel Displacement Mapping Using Hybrid Cone Approach**, *International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, *IEEE Morocco section*, ISBN 978-1-5386-0551-6, October 2017. DOI: 10.1109/ATSIP.2017.8075577

ANNEX A : GENERALITES SUR L'INFOGRAPHIE

❖ Différence entre l'infographie 2D et l'infographie 3D

Les débuts de l'infographie ont été en deux dimensions ou l'on s'intéressait plus précisément à l'exploitation du matériel informatique qui était plus modeste au profit d'un affichage graphique très simple sur un écran. Ensuite, et malgré l'arrivé de l'infographie 3D, on s'intéresse toujours à l'infographie 2D comme par exemple la création des formes géométriques 2D, la création des images fractales, le traitement d'images et l'imagerie médicale. L'infographie 2D est principalement utilisée dans des applications développées à l'origine sur des technologies traditionnelles d'impression et de dessin, telles que la typographie, la cartographie, le dessin technique, la publicité, etc. Les modèles graphiques 2D peuvent être conçu par la combinaison de plusieurs éléments tel que les modèles géométriques appelé aussi des graphiques vectoriels, des images numériques, du texte qui peut avoir des propriétés comme le style, la police, la taille, la couleur, la position, l'orientation, etc. Ces composants peuvent être modifiés et manipulés par des transformations géométriques bidimensionnelles telles que la translation, la rotation et la mise à l'échelle.

Fondamentalement, les graphiques 2D ne peuvent être animés que dans un plan a deux dimensions, c'est à dire dans deux directions : horizontalement et verticalement. Bien que les objets et les personnages puissent toujours se déplacer en diagonale sur ces axes, mais ils sont toujours confinés aux axes X et Y. Aucune profondeur ne peut être ajoutée, ce qui signifie que vous ne pouvez pas faire apparaître les objets plus près ou plus loin du spectateur. Cela s'exprime par le fait que les objets de l'infographie 2D sont représentés d'une façon plate. Des formes géométriques comme des carrés sont possibles, mais les cubes et les sphères ne peuvent pas être inclus dans ces graphiques.

Le minimalisme et la simplicité des graphiques 2D les rendent plus attrayants pour certains utilisateurs et coûtent moins chère mais sont loin de simuler le réalisme du monde réel. Le réalisme est le facteur principal qui différencie l'infographie 2D par rapport à l'infographie 3D, et cela a été concrétisé par l'introduction de l'axe Z dans la modélisation et la conception des applications de l'infographie.

Le défi majeur de l'infographie 3D est la production des scènes en trois dimensions les plus réalistes possible. Ce désir envers un réalisme parfait engendre des énormes efforts au niveau des investissements et des développements, que ce soit chez les professionnels ou chez le grand public.

❖ Système graphique

Un système graphique est l'ensembles des composants matériels et logiciels qui entrent en jeux dans la conception et la réalisation des scènes 3D. Il permet l'affichage et l'interaction avec des images, des vidéos et des graphiques sur un ordinateur ou un appareil mobile. Les systèmes graphiques modernes incluent des fonctionnalités avancées telles que la 3D, les animations, les effets de transition et la vidéo haute définition.

Un système graphique comprend généralement les composants suivants :

- Unité centrale de traitement (CPU) : c'est le composant principal d'un ordinateur qui exécute les instructions des applications graphiques.

- Mémoire vive (RAM) : stock initialement les données et les instructions des applications graphiques avant leur communication au GPU.
- Carte graphique (GPU) : c'est le composant matériel qui gère l'affichage des images et des vidéos sur l'écran. Il permet également d'accélérer les calculs liés aux graphiques 3D et il possède son propre mémoire vive (VRAM).
- Pilote de la carte graphique : c'est le logiciel qui permet au CPU de communiquer avec le GPU.
- Système d'exploitation : il gère les ressources de l'ordinateur, y compris les entrées et les sorties graphiques. Il fournit également un environnement pour les applications graphiques.
- Bibliothèques graphiques et API : les bibliothèques et API comme OpenGL et Direct3D offrent des fonctionnalités graphiques de base telles que les fonctions de dessin, de remplissage de transformation et de rendu.
- Applications graphiques : il s'agit des programmes qui utilisent les fonctionnalités graphiques pour créer des images et des vidéos.
- Moniteur : il affiche les images et les vidéos générées par le système graphique. Il peut être un écran d'ordinateur ou un téléviseur connecté à l'ordinateur via une sortie vidéo.

Il existe plusieurs types de systèmes graphiques, chacun ayant ses propres avantages et inconvénients. Les systèmes graphiques les plus courants sont :

- Les systèmes graphiques qui sont intégrés à la carte mère d'un ordinateur et qui sont généralement moins puissants mais moins chers.
- Les systèmes graphiques dédiés, qui sont des cartes graphiques séparées et installées dans un ordinateur et qui sont généralement plus puissants mais plus chères.
- Les systèmes graphiques les plus récents utilisent des technologies avancées comme l'IA, l'apprentissage automatique, les technologies ray-tracing en temps réel, etc.

Les systèmes graphiques modernes sont souvent basés sur des architectures matérielles et logicielles standards, ce qui permet aux développeurs de concevoir et de publier des applications graphiques pour une large gamme de dispositifs numériques. Les avancées technologiques continuent de faire évoluer les systèmes graphiques, avec de plus en plus de capacités pour la visualisation de données complexes et la production de contenu en temps réel.

❖ Processeurs graphiques

Les processeurs graphiques sont des circuits intégrés spécialement conçus pour gérer les tâches de traitement graphique. Ils sont généralement utilisés pour accélérer les applications graphiques et les jeux, ainsi que pour les applications de calcul scientifique intensif.

Les GPU modernes sont conçus pour traiter de grandes quantités de données en parallèle, ce qui les rend beaucoup plus rapides pour certaines tâches que les processeurs traditionnels (CPU). Les GPU peuvent gérer des milliers de processus simultanément, ce qui en fait un choix idéal pour les applications qui nécessitent des calculs complexes et en parallèles, notamment, le rendu d'images et de vidéos, les jeux, les animations, la visualisation professionnelle, les simulations scientifiques, l'intelligence artificielle et l'apprentissage automatique. Ils peuvent être intégrés à la carte mère d'un système ou installés sous forme de carte distincte et peuvent être connectés à plusieurs affichages pour des capacités d'affichage haute résolution. En utilisant les API 3D telles qu'OpenGL et Direct3D, les développeurs peuvent écrire du code qui définit comment les objets 3D doivent être dessinés et les API envoient ce code à la carte graphique pour être exécuté. Cela permet de décharger le CPU et de maximiser les performances de l'application.

Les GPU sont des composants clés pour les applications 3D modernes et sont souvent utilisés en conjonction avec les API 3D pour améliorer les performances et offrir une expérience utilisateur plus fluide et plus pratique. Dans la suite, nous allons parcourir les différentes familles de GPU les plus répondus dans le marché.

A. Les processeurs graphiques NVIDIA

NVIDIA est une entreprise américaine spécialisée dans les technologies de traitement graphique, fondée en 1993. Les GPU de NVIDIA sont utilisés dans de nombreux ordinateurs, ainsi que dans les centres d'analyse de données et les applications d'intelligence artificielle. NVIDIA propose une variété de produits matériels qui sont :

- Les GPU GeForce : ce sont des cartes graphiques conçues pour les jeux vidéo, la création de contenu et les applications d'intelligence artificielle.
- Les GPU Quadro : semblables aux cartes GeForce sauf qu'elles sont spécialement conçues pour les professionnels.
- Les GPU Tesla : ce sont des cartes graphiques hautes performance conçues pour les centres de données et les applications d'intelligence artificielle à grande échelle.
- Système embarqué Jetson : ce sont des plates-formes pour l'intelligence artificielle embarquée conçues pour les appareils connectés, les robots, les systèmes d'affichage numérique, etc.
- Appareil Shield : ce sont des appareils de divertissement qui permettent de diffuser du contenu a haute qualité, de jouer à des jeux vidéo et d'interagir avec d'autres appareils. Ils intègrent les technologie HDR, Dolby, Cloud et l'intelligence artificielle.

En plus de ses produits matériels, NVIDIA propose également une gamme complète de technologies logicielles (NVIDIA SDK) pour les développeurs, y compris des bibliothèques de traitement parallèle pour l'apprentissage profond, des outils pour le développement de jeux vidéo et des applications d'intelligence artificielle, ainsi que des solutions de visualisation de données en temps réel. En trouve entre autres :

- CUDA Toolkit : c'est un Framework qui permet aux développeurs d'utiliser les capacités de traitement parallèle des GPU pour accélérer des calculs généraux au niveaux des applications informatiques.
- GeForce Experience : c'est un logiciel qui permet aux utilisateurs de configurer et de gérer les paramètres de leurs cartes graphiques NVIDIA. Il permet également de mettre à jour les pilotes, d'optimiser les paramètres des jeux et de faire des captures d'écran et des vidéos.
- NVIDIA DRIVE : c'est un ensemble de technologies pour les véhicules autonomes qui comprend des systèmes de perception, de planification de la trajectoire et de contrôle du véhicule.
- NVIDIA CLARA : c'est une plateforme d'intelligence artificielle qui regroupe une collection de Framework, d'outils, d'applications et de modèles pré-entraînés utilisés dans le domaine de la santé.

Ces produits matériels et logiciels font partie de nombreuses solutions proposées par NVIDIA pour aider les développeurs à créer des applications plus rapides, plus puissantes et plus intelligentes en exploitant au maximum les puissances de leurs GPU. Tout au long de son histoire, NVIDIA est considérée comme l'un des leaders dans les technologies de traitement graphique et d'intelligence artificielle, avec un impact significatif sur de nombreux secteurs clés de l'industrie technologique.

B. Les processeurs graphiques AMD

AMD (Advanced Micro Devices) est une entreprise américaine de semi-conducteurs fondée en 1969. Elle développe des processeurs pour ordinateurs et des technologies associées pour les marchés des entreprises et des consommateurs.

Radeon est une marque de cartes graphiques produites par AMD et introduite pour la première fois en 2000 par ATI Technologies qui a été rachetée par AMD en 2006 et qui est devenue l'une des marques les plus reconnues et respectées dans le secteur des GPU. Les produits Radeon sont utilisés dans une variété d'applications, notamment les jeux vidéo, la conception graphique et la création de contenu multimédia. Les cartes graphiques Radeon sont considérées comme l'une des meilleures options pour les utilisateurs qui cherchent des performances élevées et une qualité graphique supérieure.

AMD produit une gamme de produits matériels pour le secteur informatique, notamment :

- Des Processeurs : AMD propose une gamme de processeurs pour les ordinateurs de bureau et portables, les stations de travail, les serveurs et les appareils mobiles. On trouve entre autres les séries Ryzen et Epyc.
- Cartes graphiques : AMD propose une série de cartes graphiques sous la marque Radeon tel que Radeon RX et Radeon Pro.
- Solutions de stockage, de mémoire et de connectivité : AMD propose également des produits de mémoire et de connectivité, y compris des contrôleurs de mémoire et des solutions de stockage NVMe.

AMD produit aussi un ensemble de produits logiciels (AMD SDK), en trouve entre autres :

- AMD APP SDK : il s'agit d'un ensemble de bibliothèques, des outils pour le développement d'applications parallèles exploitant les capacités de calcul des processeurs et des cartes graphiques AMD.
- AMD LiquidVR : cette technologie de réalité virtuelle permet aux développeurs de créer des expériences plus fluides et plus immersives en utilisant la puissance de traitement des cartes graphique AMD.
- AMD Eyefinity Technology : permet aux utilisateurs de connecter plusieurs écrans à leur carte graphique pour obtenir un environnement de travail plus grand.
- AMD Quick Stream Technology : permet une diffusion en continu et fluide en favorisant la bande passante pour les applications critiques telles que les vidéos en streaming.

C. Les processeurs graphiques ATI

ATI (Array Technologies Incorporated) était une entreprise de semi-conducteurs canadienne qui a été fondée en 1985. Elle était spécialisée dans la conception et la fabrication de produits informatiques tels que des cartes graphiques, des chipsets et des processeurs graphiques pour les ordinateurs. En 2006, ATI a été rachetée par AMD, ce qui a donné naissance à la division de produits graphiques d'AMD appelée AMD Radeon. Les produits ATI sont devenus une partie importante de la gamme de produits graphiques d'AMD et continuent d'être utilisés et appréciés par les utilisateurs dans le monde entier.

ATI était reconnue pour la production de cartes graphiques haute performance et était un choix populaire pour les jeux vidéo et les utilisateurs les plus exigeants. La technologie graphique de l'entreprise était intégrée à une large gamme de dispositifs, notamment les consoles de jeux, telles que la Xbox de Microsoft et la PlayStation de Sony. ATI était un acteur majeur de l'industrie du traitement graphique et a eu un impact significatif sur le développement du matériel graphiques.

D. Les processeurs graphiques Intel

Intel Corporation est une entreprise américaine fondée en 1968 spécialisée dans le développement des processeurs et autres puces électroniques qui sont largement utilisés dans différents équipements informatiques. La société propose également une gamme complète de produits informatiques, y compris des puces pour les réseaux, des périphériques de stockage et d'autres produits informatiques pour les entreprises et les consommateurs.

Intel Graphics désigne la technologie de traitement graphique développée par Intel Corporation. Elle est utilisée dans une large gamme de dispositifs, notamment les ordinateurs et d'autres systèmes, et est conçu pour fournir une bonne qualité visuelle et une performance pour les tâches quotidiennes telles que la navigation sur le web, la lecture de vidéos et le montage de photos.

La technologie Intel Graphics a évolué au fil des ans et comprend désormais un support pour des fonctionnalités avancées telles que l'accélération matérielle, les graphiques 3D et la lecture de vidéos haute définition. La dernière technologie Intel Xe Graphics, offre des performances élevées et prend en charge une large gamme d'applications graphiques, notamment les jeux, la création de contenu et la réalité virtuelle. Intel Graphics est connu aussi pour sa faible consommation d'énergie et est largement utilisé dans les appareils mobiles et les ordinateurs portables où l'autonomie de la batterie est un critère clé. Il est également largement pris en charge par de nombreux systèmes d'exploitation et applications logicielles, ce qui en fait une solution pratique et largement utilisée pour les besoins de base en traitement graphique.

Intel a joué un rôle important dans l'histoire de l'informatique en introduisant des processeurs plus rapides et plus puissants au fil des ans, ce qui a permis à l'industrie de l'informatique de progresser et de se développer.

❖ Interface de programmation d'application 3D (API)

Une API 3D est une interface de programmation d'application qui permet aux développeurs de créer des applications et des jeux en trois dimensions. Cela leur permet de définir des objets 3D, de les animer et de les rendre dans un environnement 3D. Les API 3D fournissent un ensemble de fonctionnalités pour les développeurs, telles que la création de scènes complexes, l'ajout d'effets spéciaux comme l'éclairage et les ombres, la gestion de la caméra et des interactions utilisateur, ainsi que la capacité de charger et de gérer des modèles 3D externes. Exemples d'API 3D populaires incluent OpenGL, DirectX et WebG. Chacune de ces API a ses propres avantages et inconvénients, mais elles partagent toutes le même objectif : offrir aux développeurs un moyen simple et efficace de créer les différentes types applications 3D.

En utilisant une API 3D, les développeurs peuvent économiser beaucoup de temps et d'efforts en se concentrant sur le développement de leur application, plutôt que sur la création d'une technologie de rendu 3D à partir de zéro. Ces API sont également souvent optimisées pour tirer parti de la puissance de traitement des ordinateurs modernes, notamment les GPU, ce qui signifie que les applications 3D créées à l'aide de ces API peuvent être plus rapides et plus performants.

A. Direct3D

Direct3D est une API développée par Microsoft en 1996 pour la création de graphiques 3D. Elle est principalement utilisée pour le développement de jeux et de simulations sur le système d'exploitation Microsoft Windows. Direct3D est un composant de la technologie DirectX de

Microsoft, qui comprend également d'autres API, telles que DirectDraw (graphiques 2D), DirectSound (audio), etc. Direct3D fournit une interface de haut niveau pour accéder aux fonctionnalités du matériel graphique, y compris le support de l'accélération matérielle, ce qui permet à l'API de tirer parti des améliorations de performance fournies par les processeurs graphiques dédiés. Direct3D a évolué au fil du temps et plusieurs versions de l'API ont été publiées, chacune apportant de nouvelles fonctionnalités et améliorations. La dernière version de l'API est Direct3D 12 sortie le 10 novembre 2020.

B. OpenGL

OpenGL (Open Graphics Library) a été développée par Silicon Graphics et a été publiée en 1992. L'API est multiplateforme, ce qui signifie que le code écrit avec OpenGL peut être exécuté sur différents systèmes d'exploitation et différents matériels graphiques. Elle permet aux développeurs de créer des graphiques riches et interactifs en utilisant une variété de primitives de dessin, telles que des points, des lignes, des triangles, et des quadrilatères. OpenGL est conçue pour être utilisée avec un grand nombre de langages de programmation, tels que C, C++, Java, Python et C#. Elle est largement adoptée par les développeurs de jeux pour créer des expériences graphiques riches tout en offrant un accès direct aux fonctionnalités matérielles. OpenGL est considérée comme un choix standard pour le développement des applications 3D en raison de sa flexibilité, de sa portabilité et de sa large communauté de développeurs qui apportent des améliorations et des corrections en continu. La dernière version de l'API est OpenGL 4.6 sortie le 31 juillet 2017.

C. OpenGL ES

OpenGL ES (Open Graphics Library for Embedded Systems) est un sous-ensemble de l'API OpenGL développée en 2003 pour les appareils intégrés tels que les smartphones, les tablettes, etc. Il fournit une interface de bas niveau et multiplateforme. Les applications écrites à l'aide d'OpenGL ES peuvent s'exécuter sur une large gamme de dispositifs, y compris iOS, Android et la plupart des consoles de jeux vidéo. OpenGL ES est une technologie importante pour l'industrie du jeu mobile, permettant aux développeurs de créer des graphiques 3D de haute qualité et reste une technologie clé pour le développement de graphiques avancés sur des appareils intégrés. Sa dernière version est OpenGL ES 3.2 sortie le 10 août 2015.

D. WebGL

WebGL est une API JavaScript développée en 2011 et qui est conçue pour les navigateurs web. Elle permet aux développeurs de créer des applications 3D sans avoir à s'appuyer sur des plug-ins ou sur d'autres logiciels tiers. L'un des avantages clés de WebGL est qu'elle permet aux développeurs de créer des graphiques riches et interactifs qui peuvent s'exécuter sur une large gamme de dispositifs, notamment les ordinateurs de bureau, les ordinateurs portables, les tablettes et les smartphones. Et comme elle est basée sur des standards web ouverts, elle peut être utilisé dans n'importe quel navigateur web qui le prend en charge. Sa dernière version est WebGL 2.0 sortie le 17 janvier 2017.

E. Vulkan

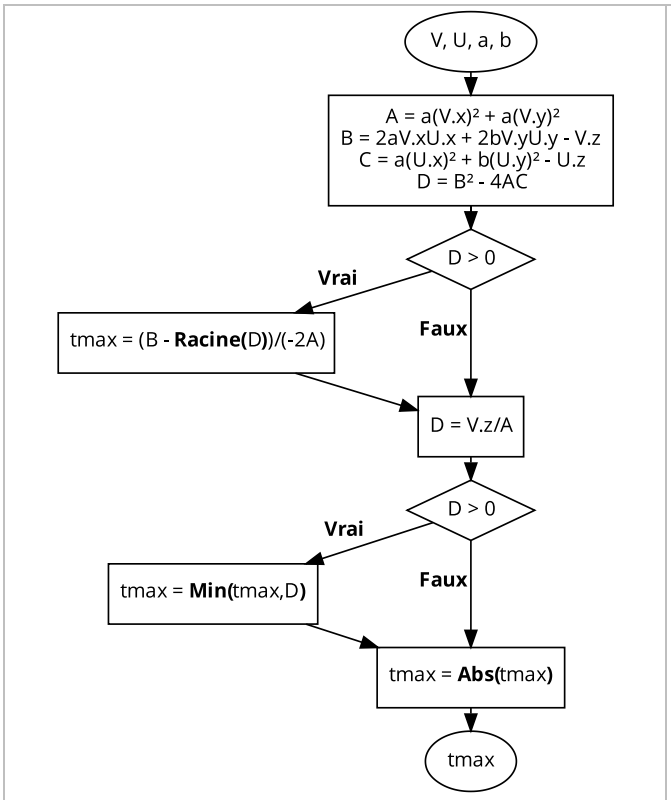
Vulkan est une API multiplateforme sortie en 2016 (supportée entre autres par AMD et NVIDIA) et conçus pour les applications et les jeux qui visent à fournir des performances graphiques optimales, les consoles de jeu, les smartphones et les tablettes. Elle a été conçue pour remplacer d'autres API graphiques plus anciennes, telles qu'OpenGL, pour les applications et les

jeux modernes qui nécessitent des performances plus élevées. Vulkan offre une surcharge minimale de CPU, ce qui signifie que le processeur n'a pas à travailler aussi dur pour gérer les tâches graphiques complexes. Cela se traduit par une utilisation plus efficace des ressources du système, ce qui permet à l'application ou au jeu de fonctionner plus rapidement et plus efficacement. Vulkan propose également une gestion améliorée de la mémoire en offrant aux développeurs la possibilité de contrôler directement les ressources graphiques, comme les textures et les images, au lieu de les laisser gérer par une couche logicielle intermédiaire. Cela se traduit par une utilisation plus efficace de la mémoire système et une réduction de la charge sur le CPU. La dernière version de l'API est Vulkan 1.3 sortie le 16 février 2023.

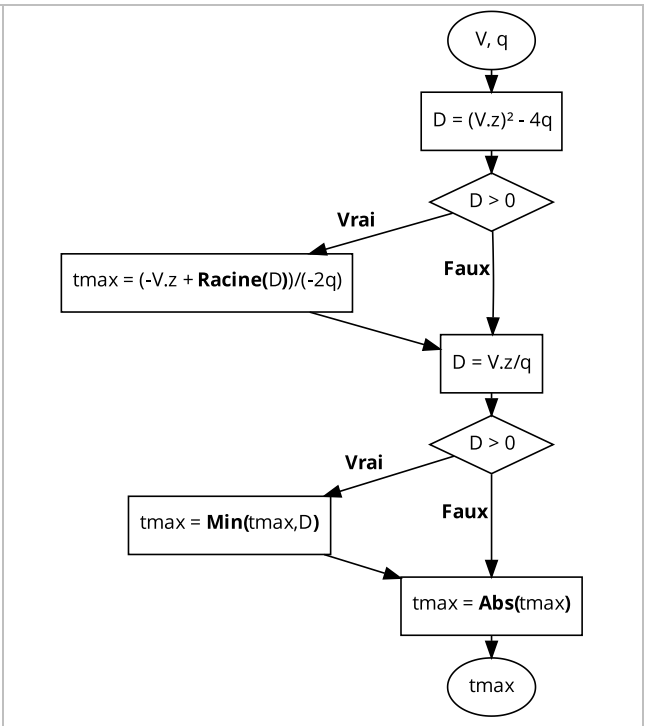
F. Metal

Metal est une API développée en 2014 par Appel pour les systèmes d'exploitation iOS et MacOs. Elle est particulièrement bien adaptée aux appareils mobiles tels que les iPhones et les iPads, où la puissance et l'efficacité au niveau des performances sont des préoccupations critiques. Metal est conçue pour être une alternative à faible surcharge aux API graphiques de haut niveau tels qu'OpenGL. Elle permet aux développeurs de mieux contrôler la manière dont le GPU gère les tâches graphiques complexes, ce qui peut entraîner une utilisation plus efficace des ressources systèmes et une amélioration des performances. La dernière version de l'API est Metal 3 sortie le 6 juillet 2022.

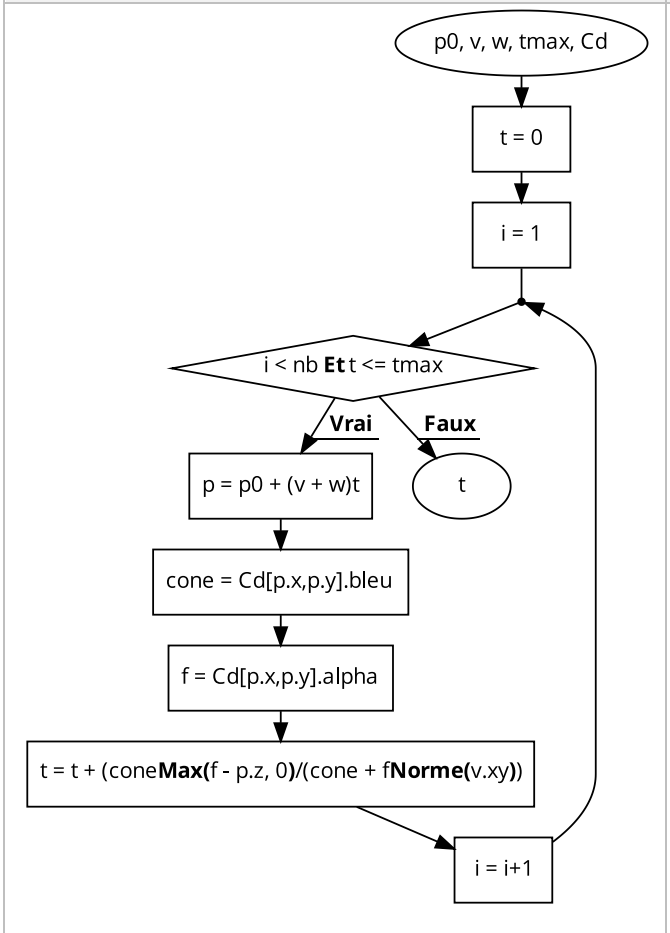
ANNEX B : ORGANIGRAMMES



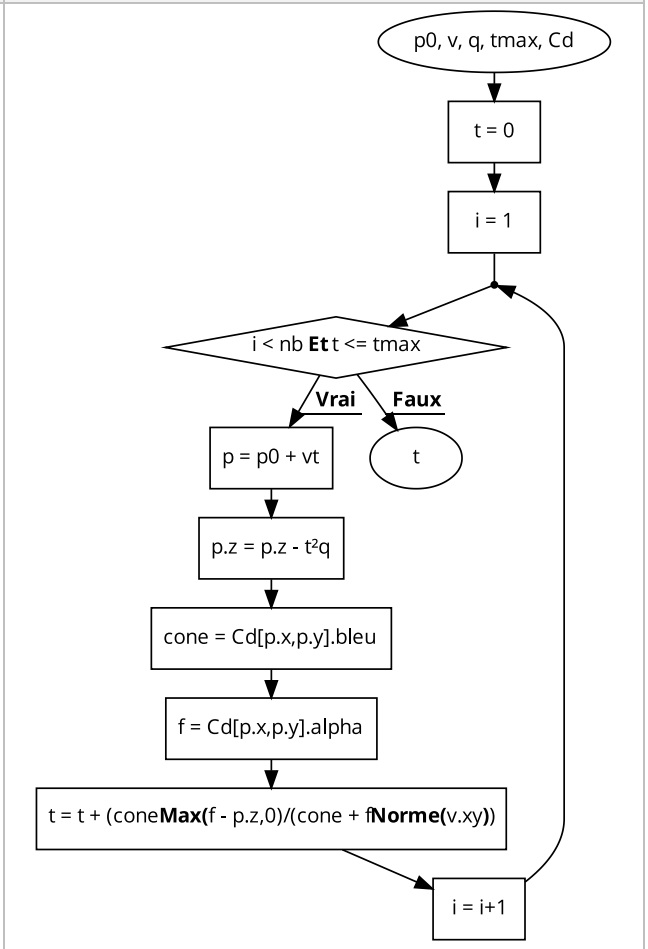
Algorithme 4.1 : tMax1



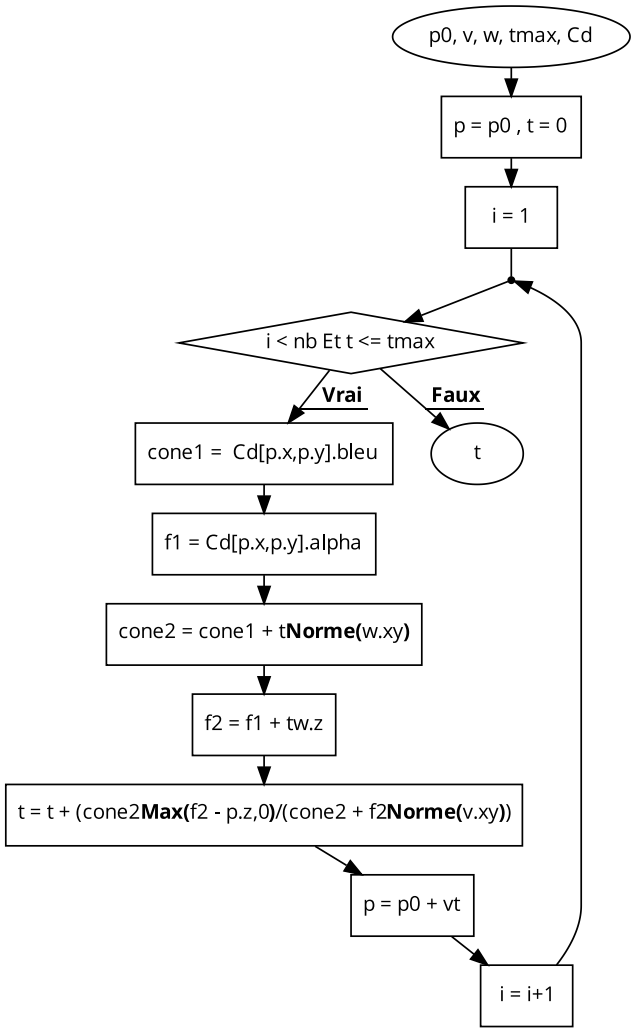
Algorithme 4.2 : tMax2



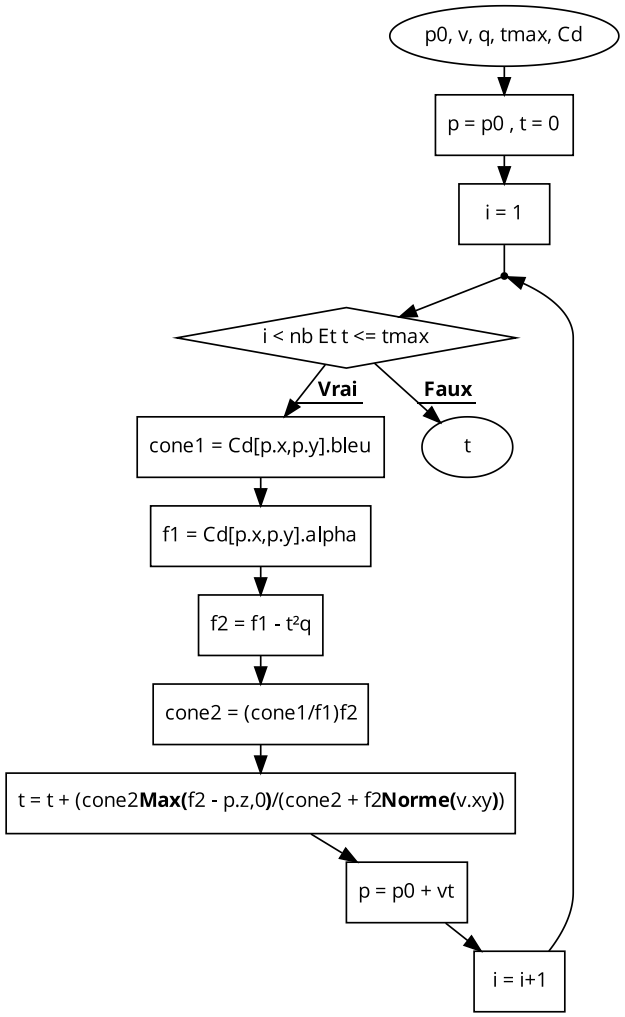
Algorithme 4.3 : RectRay1



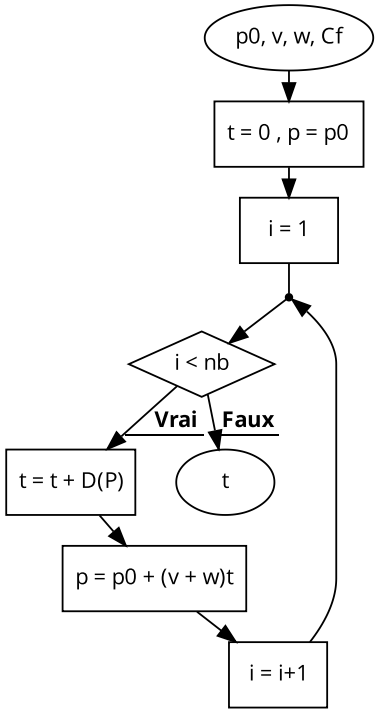
Algorithme 4.4 : RectRay2



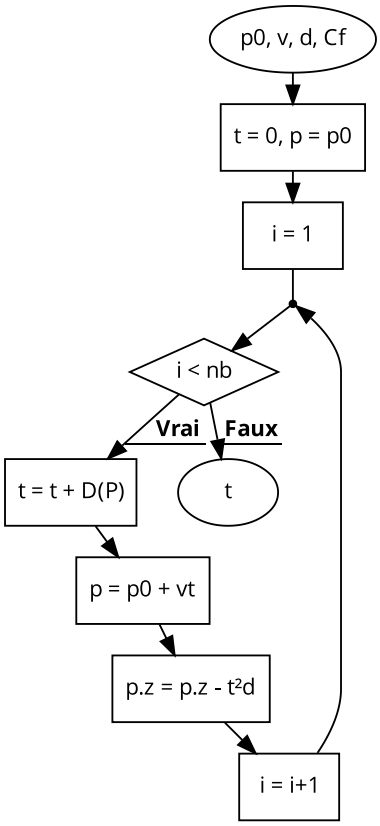
Algorithme 4.5 : RectCone1



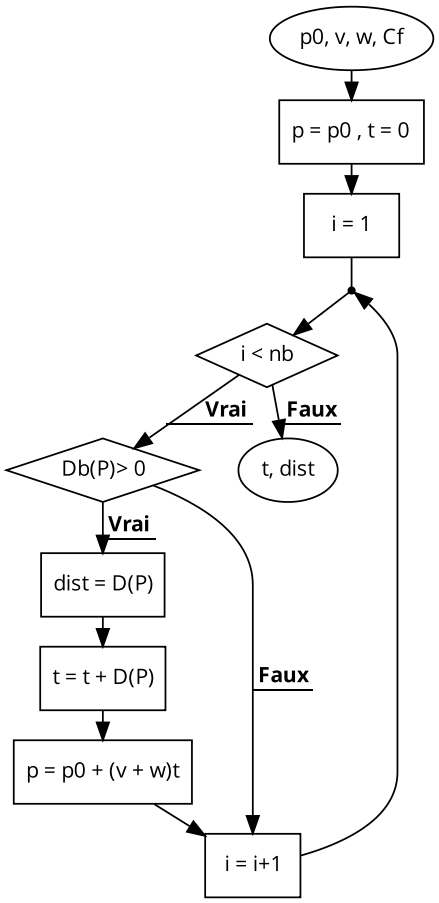
Algorithme 4.6 : RectCone2



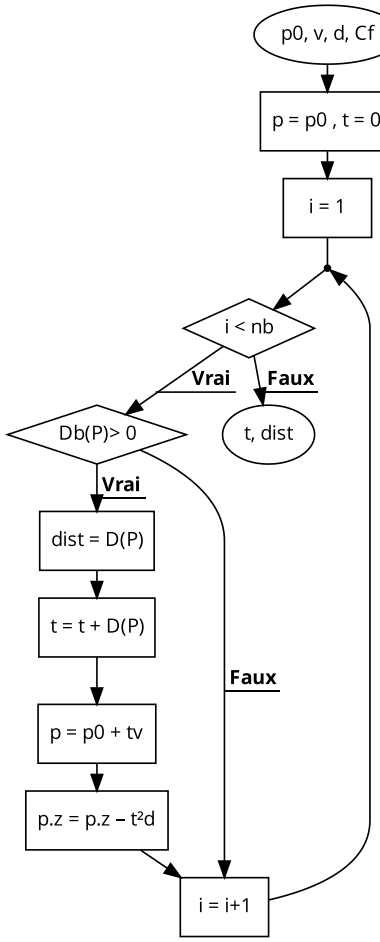
Algorithme 5.1 : ExtRectRay1



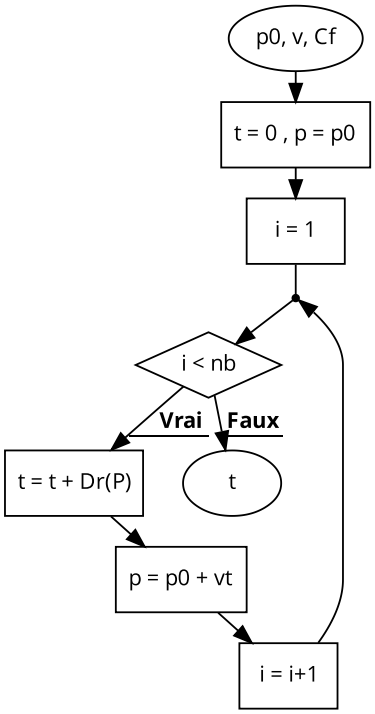
Algorithme 5.2 : ExtRectRay2



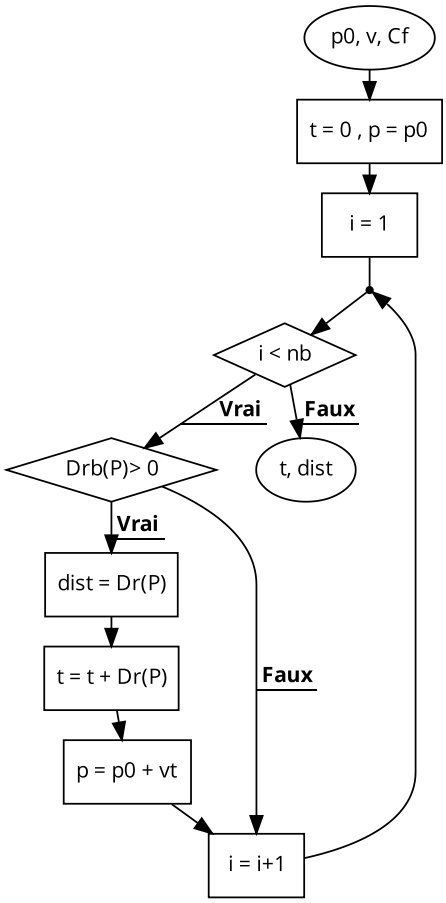
Algorithme 5.3 : ExtBiseauRectRay1



Algorithme 5.4 : ExtBiseauRectRay2



Algorithme 5.5 : ExtRectDist



Algorithme 5.6 : ExtBiseauRectDist

ANNEX C : LEXIQUE FRANÇAIS/ANGLAIS

Français	Anglais	Français	Anglais
Affichage	Display	Normale perturbée	Disturbed normal
Auto-ombrage	Self-shadowing	Phase d'ombrage	Shading phase
Occlusion	Occlusion	Phase de prétraitement	Pre-processing stage
Biseauté	Beveled	Phase de rendu	Rendering stage
Boite de forme	Shape box	Pipeline graphique	Graphical pipeline
Canal	Channel	Placage de bosselure	Bump mapping
Carte de forme	Shape map	Placage d'extrusion	Extrusion mapping
Carte de hauteur	Height map	Placage de normal	Normal mapping
Carte de profondeur	Depth map	Le placage des ombres	Shadow Mapping
Carte d'ombres	Shadow Map	Placage de parallaxe	Parallax mapping
Carte graphique	Graphics card	Placage de relief	Relief mapping
Chanfrein	Chamfer	Placage de révolution	Revolution mapping
Cônes conservatifs	Conservative cones	Placage de texture	Texture mapping
Cônes relaxés	Relaxed cones	Point d'intersection	Intersection point
Coordonnées de texture	Texture coordinates	Rendu en temps réel	Real-time rendering
Dérivées partielles	Partial derivatives	Rayon de vue	Viewing ray
Echelle de profondeur	Depth scale	Sommet	Vertex
Espace tangent	Tangent space	Surface de révolution	Revolved surface
Etendue	Outward	Surface d'extrusion	Extruded surface
Infographie	Computer graphic	Technique de traçage de cône	Cone-tracing technique
Interface de programmation d'application	Application Programming Interface	Technique de rendu	Rendering technique
Images par seconde	Frame per second	Traçage de rayons	Ray-tracing
Maillages polygonaux	Polygonal meshes	Traçage de chemins	Path-tracing
Modèle révolu	Revolved models	Vecteur de lumière	Light vector
Modélisation et rendu à base d'images	Image-based modelling and rendering	Vitesse de rendu	Rendering speed

BIBLIOGRAPHIE

[1] T. Malcolm Smith and K. C. Redmond, *Project Whirlwind the History of a Pioneer Computer*, First Edition. Digital Press, 1980.

[2] W. A. Fetter, *Computer graphics in communication*. New York, 1965.

[3] A. Careri, 'La rhétorique scientifique des jeux vidéo. Spacewar! et la translation visuelle des algorithmes', *Images re-vues*, no. 19, Dec. 2021, doi: 10.4000/imagesrevues.10058.

[4] H. R. Baer, *Videogames: In the Beginning*, First Edition. Rolenta Press, 2005.

[5] 'The TX-2 Computer and Sketchpad', *LINCOLN LABORATORY JOURNAL*, vol. VOLUME 19, no. 1, pp. 82–84, 2012.

[6] I. E. Sutherland, 'Sketchpad, a Man-Machine Graphical Communication System', Massachusetts Institute of Technology, 1963.

[7] F. N. Krull, 'The origin of computer graphics within General Motors', *IEEE Annals of the History of Computing*, vol. 16, no. 3, p. 40, 1994, doi: 10.1109/MAHC.1994.298419.

[8] I. E. Sutherland, 'A head-mounted three dimensional display', in *Proceedings of the December 9-11, 1968, fall joint computer conference, part I on - AFIPS '68 (Fall, part I)*, New York, New York, USA: ACM Press, 1968, p. 757. doi: 10.1145/1476589.1476686.

[9] E. Catmull, *Creativity, Inc.*, Talent Edition. 2020.

[10] J. F. Blinn and M. E. Newell, 'Texture and reflection in computer generated images', *ACM SIGGRAPH Computer Graphics*, vol. 10, no. 2, p. 266, 1976, doi: 10.1145/965143.563322.

[11] B. T. Phong, 'Illumination for Computer Generated Pictures', *Commun ACM*, vol. 18, no. 6, pp. 311–317, 1975, doi: 10.1145/360825.360839.

[12] J. F. Blinn, 'Models of light reflection for computer synthesized pictures', *Proceedings of the 4th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1977*, pp. 192–198, 1977, doi: 10.1145/563858.563893.

[13] J. T. Kajiya, 'The Rendering Equation', *Comput Graph (ACM)*, vol. 20, no. 4, pp. 143–150, Aug. 1986.

[14] K. J. Dana, J. J. Koenderink, K. J. Dana, S. K. Nayar, and J. J. Koenderink, 'Reflectance and Texture of Real-World Surfaces Additional Key Words and Phrases: Bidirectional reflectance distribution function, bidirec- tional texture function, " 3D texture &quot';, *ACM Trans Graph*, vol. 18, no. 1, pp. 1–34, 1999.

[15] M. A. O. Vasilescu and D. Terzopoulos, 'TensorTextures', *ACM Trans Graph*, vol. 23, no. 3, pp. 336–342, Aug. 2004, doi: 10.1145/1015706.1015725.

[16] T. Malzbender, D. Gelb, and H. Wolters, 'Polynomial texture maps', *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2001*, no. January, pp. 519–528, 2001, doi: 10.1145/383259.383320.

[17] S. F. Judice, G. A. Giraldi, and F. José Karam, 'Rendering Equation', in *Encyclopedia of Computer Graphics and Games*, Cham: Springer International Publishing, 2018, pp. 1–20. doi: 10.1007/978-3-319-08234-9_106-1.

[18] J. Arvo, 'Transfer equations in global illumination. In Global Illumination', in *SIGGRAPH 93 Course Notes*, 1993.

[19] M. Vlnas, 'Bidirectional Path Tracing', in *CESCG 2018: The 22nd Central European Seminar on Computer Graphics* , 2018.

[20] E. P. Lafortune and Y. D. Willems, 'BIDIRECTIONAL PATH TRACING', 1993.

[21] E. Veach and L. J. Guibas, 'Optimally combining sampling techniques for Monte Carlo rendering', in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques - SIGGRAPH '95*, New York, New York, USA: ACM Press, 1995, pp. 419–428. doi: 10.1145/218380.218498.

[22] D. Cline, J. Talbot, and P. Egbert, 'Energy redistribution path tracing', *ACM Trans Graph*, vol. 24, no. 3, pp. 1186–1195, Jul. 2005, doi: 10.1145/1073204.1073330.

[23] J. Chen, L. Chen, and Z. Yu, 'Accelerating path tracing rendering with Multi-GPU in Blender cycles', in *2023 25th International Conference on Advanced Communication Technology (ICACT)*, IEEE, Feb. 2023, pp. 314–318. doi: 10.23919/ICACT56868.2023.10079514.

[24] M. Kettunen, M. Manzi, M. Aittala, J. Lehtinen, F. Durand, and M. Zwicker, 'Gradient-domain path tracing', *ACM Trans Graph*, vol. 34, no. 4, pp. 1–13, Jul. 2015, doi: 10.1145/2766997.

[25] X. Pan, V. Arellano, and A. Jarabo, 'Transient instant radiosity for efficient time-resolved global illumination', *Comput Graph*, vol. 83, pp. 107–113, Oct. 2019, doi: 10.1016/j.cag.2019.07.009.

[26] S. E. Chen, H. E. Rushmeier, G. Miller, and D. Turner, 'A progressive multi-pass method for global illumination', in *Proceedings of the 18th annual conference on Computer graphics and interactive techniques - SIGGRAPH '91*, New York, New York, USA: ACM Press, 1991, pp. 165–174. doi: 10.1145/122718.122737.

[27] A. Keller, 'A Quasi-Monte Carlo Algorithm for the Global Illumination Problem in the Radiosity Setting', 1995, pp. 239–251. doi: 10.1007/978-1-4612-2552-2_15.

[28] H. R. Zatz, 'Galerkin radiosity', in *Proceedings of the 20th annual conference on Computer graphics and interactive techniques - SIGGRAPH '93*, New York, New York, USA: ACM Press, 1993, pp. 213–220. doi: 10.1145/166117.166145.

[29] G. Nichols, J. Shopf, and C. Wyman, 'Hierarchical Image-Space Radiosity for Interactive Global Illumination', *Computer Graphics Forum*, vol. 28, no. 4, pp. 1141–1149, Jun. 2009, doi: 10.1111/j.1467-8659.2009.01491.x.

[30] D. P. Greenberg, M. F. Cohen, and K. E. Torrance, 'Radiosity: A method for computing global illumination', *Vis Comput*, vol. 2, no. 5, pp. 291–297, Sep. 1986, doi: 10.1007/BF02020429.

[31] C. Kang, L. Wang, Y. Xu, and X. Meng, 'A survey of photon mapping state-of-the-art research and future challenges', *Frontiers of Information Technology & Electronic Engineering*, vol. 17, no. 3, pp. 185–199, Mar. 2016, doi: 10.1631/FITEE.1500251.

[32] P. H. Christensen, 'Faster Photon Map Global Illumination', *Journal of Graphics Tools*, vol. 4, no. 3, pp. 1–10, Jan. 1999, doi: 10.1080/10867651.1999.10487505.

[33] M. McGuire and D. Luebke, 'Hardware-accelerated global illumination by image space photon mapping', in *Proceedings of the Conference on High Performance Graphics 2009*, New York, NY, USA: ACM, Aug. 2009, pp. 77–89. doi: 10.1145/1572769.1572783.

[34] T. Hachisuka, S. Ogaki, and H. W. Jensen, 'Progressive photon mapping', in *ACM SIGGRAPH Asia 2008 papers*, New York, NY, USA: ACM, Dec. 2008, pp. 1–8. doi: 10.1145/1457515.1409083.

[35] T. Hachisuka and H. W. Jensen, 'Stochastic progressive photon mapping', in *ACM SIGGRAPH Asia 2009 papers*, New York, NY, USA: ACM, Dec. 2009, pp. 1–8. doi: 10.1145/1661412.1618487.

[36] H. W. Jensen, 'Global Illumination using Photon Maps', 1996, pp. 21–30. doi: 10.1007/978-3-7091-7484-5_3.

[37] Catmull and E. Earl, 'A subdivision algorithm for computer display of curved surfaces.' The University of Utah, 1974. [Online]. Available: https://dl.acm.org/citation.cfm?id=907242

[38] J. F. Blinn, 'Simulation of wrinkled surfaces', *ACM SIGGRAPH Computer Graphics*, vol. 12, no. 3, pp. 286–292, 1978, doi: 10.1145/965139.507101.

[39] M. Peercy, J. Airey, and B. Cabral, 'Efficient bump mapping hardware', *Proceedings of the 24th annual conference on Computer graphics and interactive techniques - SIGGRAPH '97*, pp. 303–306, 1997, doi: 10.1145/258734.258873.

[40] J. Gath, 'Derivation of the Tangent Space Matrix', pp. 1–7, 2003.

[41] R. L. Cook, 'SHADE TREES.', *Computer Graphics (ACM)*, vol. 18, no. 3. 1984.

[42] A. Lee, H. Moreton, and H. Hoppe, 'Displaced subdivision surfaces', in *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, 2000.

[43] S. Gumhold and T. Hüttner, 'Multiresolution rendering with displacement mapping', pp. 55–66, 1999, doi: 10.1145/311534.311578.

[44] M. Doggett and J. Hirche, 'Adaptive view dependent tessellation of displacement maps', *Proceedings of the SIGGRAPH/Eurographics Workshop on Graphics Hardware*, pp. 59–66, 2000, doi: 10.1145/346876.348220.

[45] J. W. Patterson, S. G. Hoggar, and J. R. Logie, 'Inverse Displacement Mapping', *Computer Graphics Forum*, vol. 10, no. 2, pp. 129–139, 1991, doi: 10.1111/1467-8659.1020129.

[46] A. Halli, A. Saaidi, K. Satori, and H. Tairi, 'Per-Pixel Displacement Mapping Using Cone Tracing', *International Review on Computers and Software*, vol. 3, no. 3, pp. 1–11, 2008.

[47] L. Szirmay-Kalos and T. Umenhoffer, 'Displacement Mapping on the GPU — State of the Art', *Computer Graphics Forum*, vol. 27, no. 6, pp. 1567–1592, Nov. 2008, doi: 10.1111/J.1467-8659.2007.01108.X.

[48] A. RAGRAGUI, 'Rendu en temps réel des modèles d'extrusion et de révolution avec des rides dynamiques et reliefs réalistes', Université Sidi Mohamed Ben Abdellah, Fès, 2020.

[49] T. Kaneko *et al.*, 'Detailed Shape Representation with Parallax Mapping', *In Proceedings of the ICAT 2001*, pp. 205–208, 2001.

[50] T. Welsh, 'Parallax mapping with offset limiting: A per-pixel approximation of uneven surfaces', *Infiscape Corporation*, pp. 1–9, 2004, [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Parallax+Mapping+with+Offset+Limiting+:+A+Per+?+Pixel+Approximation+of+Uneven+Surfaces#0

[51] M. Premecz, 'Iterative Parallax Mapping with Slope Information', *Central European Seminar on Computer Graphics*, no. 1, pp. 1–8, 2006.

[52] M. Mcguire and M. Mcguire, 'Steep Parallax Mapping', *I3D 2005 Poster*, p. 2005, 2005, [Online]. Available: http://www.cs.brown.edu/research/graphics/games/SteepParallax/index.html

[53] N. Tatarchuk, 'Dynamic parallax occlusion mapping with approximate soft shadows', *Proceedings of the 2006 symposium on Interactive 3D graphics and games - SI3D '06*, p. 63, 2006, doi: 10.1145/1111411.1111423.

[54] N. Tatarchuk, 'Practical dynamic parallax occlusion mapping', *ACM SIGGRAPH 2005 Sketches on - SIGGRAPH '05*, p. 106, 2005, doi: 10.1145/1187112.1187240.

[55] Z. Brawlley and N. Tatarchuk, 'Parallax Occlusion Mapping: Self Shadowing, Perspective-Correct Bump Mapping Using Reverse Height Map Tracing', in *ShaderX3*, 2004, pp. 135–154.

[56] N. Tatarchuk and Natalya, 'Practical parallax occlusion mapping with approximate soft shadows for detailed surface rendering', in *ACM SIGGRAPH 2006 Courses on - SIGGRAPH '06*, ACM Press, 2006, p. 81. doi: 10.1145/1185657.1185830.

[57] K. Yerex and M. Jagersand, 'Displacement Mapping with Ray-casting in Hardware', *Siggraph 2004 Sketches*, p. 2000, 2004, doi: 10.1145/1186223.1186410.

[58] M. S. S. Pattanaik E. Risser, 'Interval Mapping', *Symposium on Interactive 3D Graphics and Games (I3D)*, 2006, doi: 10.1002/gepi.1370120632.

[59] F. Policarpo, M. M. Oliveira, and J. L. D. Comba, 'Real-time relief mapping on arbitrary polygonal surfaces', *ACM SIGGRAPH 2005 Papers on - SIGGRAPH '05*, p. 935, 2005, doi: 10.1145/1186822.1073292.

[60] M. M. Oliveira, G. Bishop, and D. McAllister, 'Relief texture mapping', *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00*, pp. 359–368, 2000, doi: 10.1145/344779.344947.

[61] W. Donnelly, 'Per-Pixel Displacement Mapping with Distance Functions', *GPU Gems 2*, pp. 123–137, 2005, [Online]. Available: http://http.download.nvidia.com/developer/GPU_Gems_2/GPU_Gems2_ch08.pdf

[62] J. C. Hart, 'Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces', *Visual Computer*, vol. 12, no. 10, pp. 527–545, 1996, doi: 10.1007/s003710050084.

[63] L. Baboud and X. Decoret, 'Rendering geometry with relief textures', *Proceedings of the 2006 Conference on Graphics Interface*, pp. 195–201, 2006, [Online]. Available: http://portal.acm.org/citation.cfm?id=1143079.1143112

[64] D. W. Paglieroni, 'Height Distributional Distance Transform Methods for Height Field Ray Tracing', vol. 13, no. 4, 1994.

[65] F. Policarpo and M. M. Oliveira, 'Relaxed cone stepping for relief mapping', *GPU Gems 3*, no. Dummer, pp. 409–428, 2007.

[66] J. Dummer, 'Cone step mapping: An iterative ray-heightfield intersection algorithm', *Dostupno na http://www. lonesock. net/files/*, 2006, [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Cone+Step+Mapping:+An+iterative+ray-heightfield+intersection+algorithm#0

[67] G. Valasek and R. Bán, 'Quick Cone Map Generation on the GPU', in *Eurographics 2022 - Short Papers*, The Eurographics Association, 2022.

[68] A. Kolb and C. Rezk-Salama, 'Efcient Empty Space Skipping for Per-Pixel Displacement Mapping', *undefined*, 2005.

[69] K. Oh, H. Ki, and C. H. Lee, 'Pyramidal displacement mapping: A GPU based artifacts-free ray tracing through an image pyramid', *Proceedings of the ACM Symposium on Virtual Reality Software and Technology, VRST*, pp. 75–82, 2006, doi: 10.1145/1180495.1180511.

[70] H. Ki, 'Accurate Per-Pixel Displacement Mapping using a Pyramid Structure', *Building*, 2007.

[71] A. Tevs, I. Ihrke, and H. P. Seidel, 'Maximum mipmaps for fast, accurate, and scalable dynamic height field rendering', *Proceedings of the Symposium on Interactive 3D Graphics and Games, I3D 2008*, pp. 183–190, 2008, doi: 10.1145/1342250.1342279.

[72] D. Cohen and A. Shaked, 'Photo-Realistic Imaging of Digital Terrains', *Computer Graphics Forum*, vol. 12, no. 3, pp. 363–373, Aug. 1993, doi: 10.1111/1467-8659.1230363.

[73] M. Drobot, 'Quadtree Displacement Mapping with Height Blending', in *GPU Pro 360 Guide to Rendering*, 1st Edition., 2018.

[74] A. Halli, A. Saaidi, K. Satori, and H. Tairi, 'Per-Pixel Extrusion Mapping', *International Journal of Computer Science and Network Security*, vol. 9, no. 3. p. 118, 2009. [Online]. Available: https://pdfs.semanticscholar.org/945b/0efa92d69fee8a7c604962e04cf647917bad.pdf

[75] P. E. Danielsson, 'Euclidean distance mapping', *Computer Graphics and Image Processing*, vol. 14, no. 3, pp. 227–248, 1980, doi: 10.1016/0146-664X(80)90054-4.

[76] R. Fabbri, L. D. F. Costa, J. C. Torelli, and O. M. Bruno, '2D Euclidean distance transform algorithms', *ACM Comput Surv*, vol. 40, no. 1, pp. 1–44, 2008, doi: 10.1145/1322432.1322434.

[77] S. Gustavson and R. Strand, 'Anti-aliased Euclidean distance transform', *Pattern Recognit Lett*, vol. 32, no. 2, pp. 252–257, 2011, doi: 10.1016/j.patrec.2010.08.010.

[78] A. Halli, A. Saaidi, K. Satori, and H. Tairi, 'Extrusion and revolution mapping', *ACM Trans Graph*, vol. 29, no. 5, pp. 1–14, Nov. 2010, doi: 10.1145/1857907.1857908.

[79] A. Ragragui, A. O. Chahdi, A. Halli, and K. Satori, 'Per-Pixel Extrusion Mapping: The correction of the intersection point between the extrusion geometry and the viewing ray', in *2017 Intelligent Systems and Computer Vision, ISCV 2017*, 2017. doi: 10.1109/ISACV.2017.8054957.

[80] A. Ragragui, A. Ouazzani Chahdi, A. Halli, and K. Satori, 'Image-based extrusion with realistic surface wrinkles', *J Comput Des Eng*, vol. 7, no. 1, pp. 30–43, Nov. 2020, doi: 10.1093/jcde/qwaa004.

[81] A. Ragragui, A. O. Chahdi, A. Halli, and K. Satori, 'Per-pixel revolution mapping with rectification of the texture projection', in *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*, IEEE, Nov. 2018, pp. 1–6. doi: 10.1109/ISACV.2018.8354056.

[82] A. Ragragui, A. O. Chahdi, A. Halli, K. Satori, and H. El Moubtahij, 'The extensions of revolution-bump mapping', *Journal of Graphic Engineering and Design*, vol. 13, no. 1, pp. 21–31, 2022, doi: 10.24867/JGED-2022-1-021.

[83] S. P. Jean, 'A Survey of Methods for Recovering Quadrics in Triangle Meshes', *ACM Comput Surv*, vol. 34, no. 2, pp. 211–262, 2002, doi: 10.1145/508352.508354.

[84] M. M. Oliveira and F. Policarpo, 'An Efficient Representation for Surface Details', vol. 55, no. 51, pp. 1–8, 2005, doi: http://www.inf.ufrgs.br/~oliveira/RTM.html.

[85] Y. C. Chen and C. F. Chang, 'A prism-free method for silhouette rendering in inverse displacement mapping', *Computer Graphics Forum*, vol. 27, no. 7, pp. 1929–1936, 2008, doi: 10.1111/j.1467-8659.2008.01341.x.

[86] K.-G. Na and M.-R. Jung, 'Curved Ray-Casting for Displacement Mapping in the GPU', *Advances in Multimedia Modeling*. Springer Berlin Heidelberg, pp. 348–357, 2008. doi: 10.1007/978-3-540-77409-9_33.

[87] J. Hirche, A. Ehlert, M. Doggett, and S. Guthe, 'Hardware accelerated per-pixel displacement mapping', *Proceedings - Graphics Interface*, pp. 153–160, 2004.

[88] P. Shirley and A. Tuchman, 'A polygonal approximation to direct scalar volume rendering', in *Proceedings of the 1990 workshop on Volume visualization - VVS '90*, ACM Press, Nov. 1990, pp. 63–70. doi: 10.1145/99307.99322.

[89] S. D. Porumbescu, B. Budge, L. Feng, and K. I. Joy, 'Shell maps', *ACM Trans Graph*, vol. 24, no. 3, pp. 626–633, Nov. 2005, doi: 10.1145/1073204.1073239.

[90] J.-F. Dufort, L. Leblanc, and P. Poulin, 'Interactive Rendering of Meso-structure Surface Details using Semi-transparent 3D Textures', 2005.

[91] S. Jeschke, S. Mantler, and M. Wimmer, 'Interactive Smooth and Curved Shell Mapping', 2007.

[92] L. Wang *et al.*, 'View-dependent displacement mapping', in *ACM SIGGRAPH 2003 Papers on - SIGGRAPH '03*, ACM Press, 2003, p. 334. doi: 10.1145/1201775.882272.

[93] X. Wang, X. Tong, S. Lin, S. Hu, B. Guo, and H.-Y. Shum, 'Generalized displacement maps', *Proceedings of the Fifteenth Eurographics conference on Rendering Techniques*, pp. 227–233, 2004, doi: 10.2312/egwr/egsr04/227-233.

[94] L. Williams, 'Casting curved shadows on curved surfaces', *ACM SIGGRAPH Computer Graphics*, vol. 12, no. 3, pp. 270–274, Aug. 1987.

[95] P. Timokhin and M. Mikhaylyuk, 'Reliable GPU-Based Methods and Algorithms of Implementation Dynamic Relief Shadows in Virtual Environment Systems', in *31st International Conference on Computer Graphics and Vision*, CEUR Workshop Proceedings, Sep. 2021.

[96] N. L. Max, 'Horizon mapping: shadows for bump-mapped surfaces', *Vis Comput*, vol. 4, no. 2, pp. 109–117, Nov. 1988, doi: 10.1007/BF01905562.

[97] P.-P. J. Sloan and M. F. Cohen, 'Interactive Horizon Mapping', pp. 281–286, 2000, doi: 10.1007/978-3-7091-6303-0_25.

[98] W. Heidrich, K. Daubert, J. Kautz, and H. P. Seidel, 'Illuminating micro geometry based on precomputed visibility', *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, pp. 455–464, 2000, doi: 10.1145/344779.344984.

[99] A. Halli, K. Satori, H. Tairi, and A. Saaidi, 'Relief Mapping with Real Time Depth Scaling Support', in *Information and Communication Technologies International Symposium*, Fez: IEEE, Apr. 2007, pp. 476–479.