April 15, 2025

Chapitre: Infographie et Science des Données: Fondements Mathématiques et Pipeline Graphique

0.1 Introduction

L'infographie est un domaine vaste et dynamique au croisement de l'informatique, des mathématiques, de la physique et de l'art. Elle concerne la création, la manipulation et l'affichage d'images et de modèles numériques.

0.1.1 Définition

Definition 0.1. L'infographie peut être définie comme l'application de l'informatique à la création, au traitement, et à l'exploitation des images numériques . C'est un terme valise formé à partir des mots INFOrmatique et GRAPHIque, dont l'appellation a été déposée par la société Benson en 1974.

Ce domaine est intrinsèquement interdisciplinaire, faisant appel à :

- Physique (modélisation de la lumière)
- Mathématiques (géométrie, algèbre linéaire)
- Perception humaine (couleurs, formes)
- Interaction homme-machine (interfaces)
- Ingénierie (matériel graphique, logiciels)
- Conception graphique et art (esthétique, communication visuelle)

Conceptuellement, l'infographie prend en entrée une description d'une scène (géométrie, matériaux, lumière, caméra) et produit une image en sortie. C'est le processus inverse de la vision par ordinateur (Computer Vision), qui analyse une image pour en extraire une description ou une compréhension.

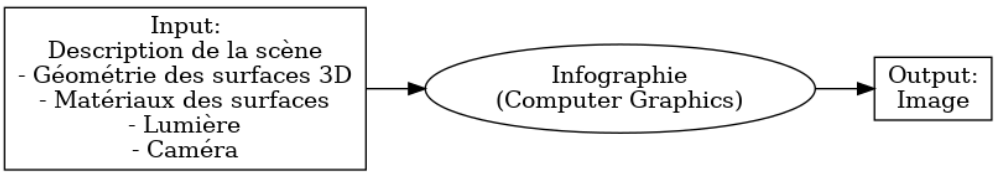


Figure 1: Processus conceptuel de l'infographie.

Une scène 3D typique est composée de plusieurs éléments clés :

- Objets 3D : La géométrie des formes présentes dans la scène.
- Matériaux : Les propriétés de surface des objets (couleur, texture, réflectivité).
- Lumières : Les sources éclairant la scène.

- Caméra : Le point de vue à partir duquel la scène est observée.

L'image de synthèse finale résulte de l'interaction entre les sources de lumière, les objets et leurs matériaux, vue depuis la caméra. La **géométrie** est l'élément constitutif essentiel, et l'organisation spatiale des objets forme le **modèle géométrique** de la scène virtuelle.

En infographie, le terme **modèle** peut désigner :

- Un **modèle géométrique** : La représentation d'un objet avec ses attributs (couleur, texture, réflectance, etc.).
- Un **modèle mathématique** : La description d'un processus physique ou informatique (par exemple, le calcul de la réflexion de la lumière sur une surface).

0.1.2 Évolution Historique

L'infographie a une riche histoire marquée par des avancées logicielles et matérielles.

Les débuts interactifs :

- **1963 : Ivan Sutherland - SKETCHPAD**. Considéré comme le pionnier de l'infographie interactive. Sketchpad était le premier système complet permettant de sélectionner, pointer, dessiner et éditer graphiquement sur un écran. Il introduisait des concepts clés comme les structures de données graphiques, les algorithmes nécessaires, la modélisation hiérarchique et les menus contextuels (pop-up).

Affichage :

- **Écrans vectoriels :** Les premiers écrans traçaient des lignes directement.
  - 1963 : Oscilloscope modifié (utilisé pour Sketchpad).
  - 1974 : "Picture System" d'Evans et Sutherland.
- **Écrans raster :** Affichent une grille de pixels (bitmap).
  - 1975 : Buffer de frames (mémoire vidéo) par Evans et Sutherland.
  - Années 1980 : Ordinateurs personnels basés sur les bitmaps (ex: Apple Macintosh, 1984).
  - Années 1990 : Écrans LCD (Liquid-Crystal Displays) sur les ordinateurs portables.
  - Années 2000 : Appareils photo numériques, projecteurs à micro-miroirs. Les écrans CRT (Cathode Ray Tube) sont progressivement remplacés par les LCD.
- **Autres technologies :** Stéréo, casques de Réalité Virtuelle, interfaces tactiles, haptiques, audio 3D.

Entrées (Inputs) :

- **2D :** Stylo lumineux, tablette graphique, souris, joystick, trackball, écran tactile.
- **3D :** Traqueurs de mouvement 3D, systèmes multi-caméras, télémètres actifs.
- **Autres :** Gants de données (tactiles), reconnaissance vocale, reconnaissance gestuelle.

Rendu (Simulation de la lumière) :

- **Années 1960 - Visibilité :** Résolution du problème des lignes et surfaces cachées.
  - Roberts (1963), Appel (1967) : Algorithmes de lignes cachées.
  - Warnock (1969), Watkins (1970) : Algorithmes de surfaces cachées.
  - Sutherland (1974) : Tri par visibilité.

- **Années 1970 - Graphiques Raster :** Premiers modèles d'ombrage locaux.
  - Gouraud (1971) : Ombrage lisse (interpolation des couleurs).
  - Phong (1974) : Modèle d'illumination spéculaire.
  - Blinn (1974) : Mapping de texture, surfaces courbes.
  - Catmull (1974) : Z-buffer (tampon de profondeur) pour la visibilité.
  - Crow (1977) : Techniques d'anti-aliasing (anticrénelage).
- **Début Années 1980 - Illumination globale :** Simulation plus réaliste de la lumière.
  - Whitted (1980) : Ray tracing (lancer de rayons) pour réflexions et réfractions.
  - Goral et al. (1984), Cohen (1985) : Radiosité pour les inter-réflexions diffuses.
  - Kajiya (1986) : Équation de rendu (formalisation générale).
- **Fin Années 1980 - Photoréalisme :** Techniques avancées pour le réalisme.
  - Cook (1984) : Arbres d'ombrage (Shading trees).
  - Perlin (1985) : Bruit de Perlin (textures procédurales), langage de shading.
  - Hanrahan & Lawson (1990) : RenderMan (standard de rendu).
- **Début Années 1990 - Rendu Non-Photoréaliste (NPR) :** Styles artistiques.
  - Drebin et al. (1988), Levoy (1988) : Rendu de volumes (visualisation médicale).
  - Haeberli (1990) : Programmes de peinture impressionniste.
  - Salesin et al. (1994-) : Illustration automatique à l'encre et au stylo.
  - Meier (1996) : Rendu de peinture (simulation de coups de pinceau).
- **Fin Années 1990 - Rendu Basé sur Images (IBR) :** Utilisation de photographies.
  - Chen & Williams (1993) : Interpolation de points de vue.
  - McMillan & Bishop (1995) : Modélisation plénoptique.
  - Levoy & Hanrahan (1996) : Rendu des champs lumineux.

Programmation et Matériel :

- **Début Années 1980 - Cartes Graphiques :** Matériel dédié.
  - Clark (1979) : Premier processeur programmable dédié au graphisme 3D (Geometry Engine).
  - 1982 : Création de Silicon Graphics (SGI), Adobe et AutoDesk.
  - 1987 : Première carte graphique grand public.
  - 1992 : OpenGL 1.0 (API graphique standard).
  - 1993 : Création de NVIDIA.
  - 1994 : VRML (Virtual Reality Modeling Language).
  - 1996 : DirectX de Microsoft.
  - 1997 : Java3D de Sun Microsystems.
  - 2007 : OpenGL 3.0 (évolution majeure avec shaders programmables).

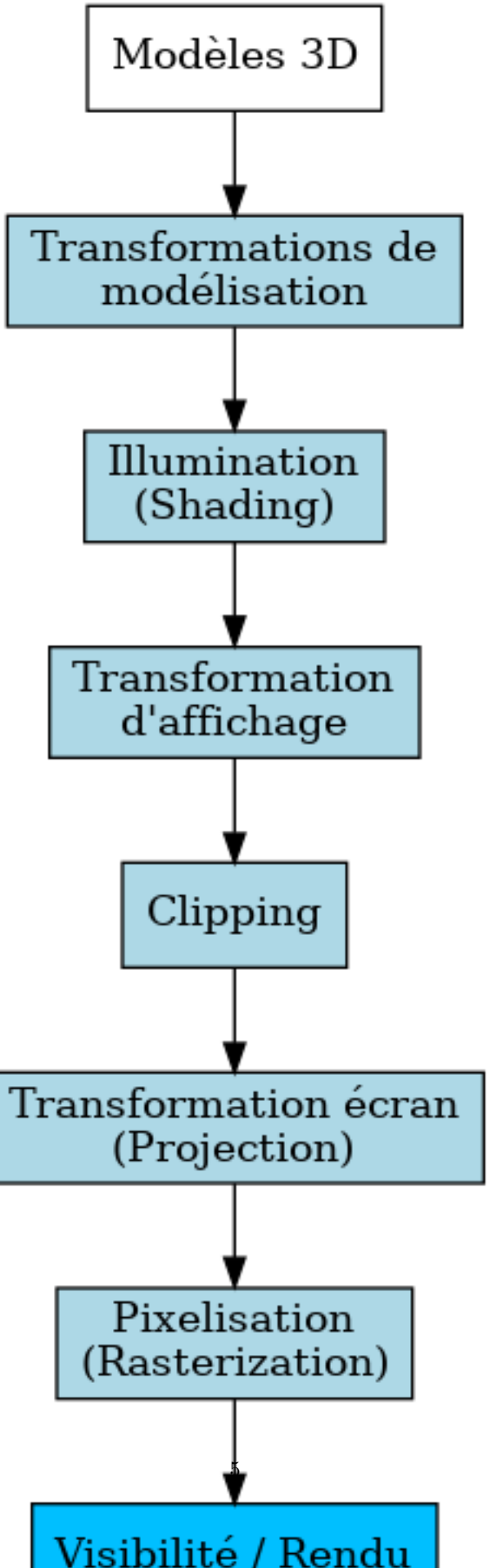
0.1.3 Applications

L'infographie trouve des applications dans de nombreux domaines :

- **Science des Données :** Visualisation de données complexes (médicales, géophysiques, biologiques, dynamique des fluides - CFD) pour l'analyse et la compréhension.
- **Divertissement :**
  - Films d'animation (ex: Shrek).
  - Effets spéciaux (VFX) dans les films (ex: simulation d'explosions, personnages numériques).
  - Jeux vidéo (temps réel, interaction).
- **Art & Design :** Création artistique numérique, design graphique, illustration.
- **Design Industriel & Ingénierie Assistée par Ordinateur (CAE) :** Conception et visualisation de produits (ex: voitures), simulations techniques.
- **Supervision & Téléopération :** Contrôle à distance de systèmes (ex: robots chirurgicaux), interfaces de supervision.
- **Simulateurs (offline) :**
  - Entraînement (conduite, pilotage).
  - Jeux de simulation.
  - Reconstruction virtuelle (ex: bâtiments historiques).
- **Navigation :** Systèmes de cartographie 3D, aide à la navigation.
- **Communications :** Présentations visuelles, supports de communication (ex: conférences comme SIGGRAPH).

0.2 I. Le Pipeline Graphique

Le pipeline graphique est une séquence d'étapes conceptuelles utilisées pour transformer une description de scène 3D en une image 2D affichable sur un écran. Chaque primitive géométrique (souvent des triangles) traverse ces étapes.


Modèles 3D
Transformations de modélisation
Illumination (Shading)
Transformation d'affichage
Clipping
Transformation écran (Projection)
Pixelisation (Rasterization)
Visibilité / Rendu

0.2.1 1. Transformations de Modélisation

- **Objectif :** Positionner et orienter chaque objet 3D dans un espace commun appelé "monde" (World Space).
- **Processus :** Applique des transformations (translation, rotation, mise à l'échelle) aux coordonnées locales de chaque objet (Object Space) pour les placer dans le repère global (World Space).

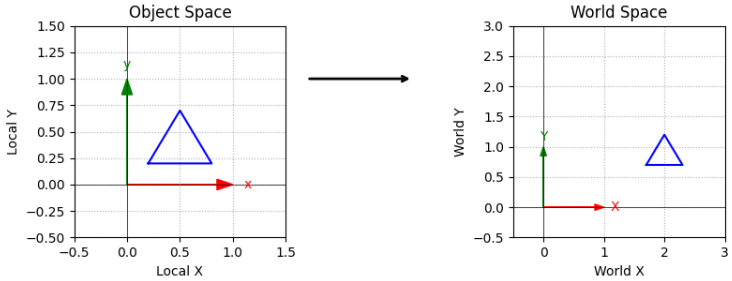


Figure 3: Passage de l'espace objet (Object Space) à l'espace monde (World Space).

0.2.2 2. Illumination (Shading)

- **Objectif :** Calculer la couleur de chaque point visible des objets en fonction des matériaux, des sources de lumière et de la position de l'observateur.
- **Processus :** Les modèles d'illumination (souvent locaux à ce stade, sans ombres portées complexes) calculent l'interaction de la lumière avec la surface. Exemples de modèles : diffus (Lambert), spéculaire (Phong, Blinn-Phong), ambiant. Les calculs sont effectués par primitive (triangle) ou par sommet.

0.2.3 3. Transformation d'Affichage (Viewing Transformation)

- **Objectif :** Transformer les coordonnées du monde (World Space) vers l'espace de la caméra (Eye Space ou View Space).
- **Processus :** Place la caméra à l'origine et aligne sa direction de vue avec un axe (souvent -z). Les coordonnées des objets sont recalculées par rapport à ce nouveau repère.

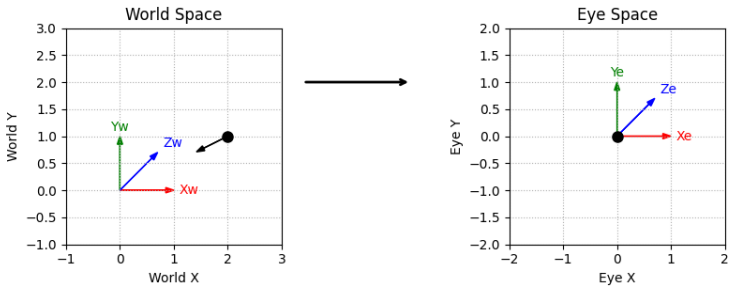


Figure 4: Passage de l'espace monde (World Space) à l'espace caméra (Eye Space).

0.2.4 4. Clipping

- **Objectif :** Supprimer les parties de la scène qui sont en dehors du volume de vision défini par la caméra (View Frustum).
- **Processus :** Les primitives (triangles) sont coupées ou rejetées si elles se trouvent en dehors des six plans définissant le frustum (proche, lointain, gauche, droite, haut, bas). Les coordonnées sont souvent normalisées dans cette étape ou la suivante pour former les Coordonnées Normalisées de Périphérique (NDC - Normalized Device Coordinates), généralement dans un cube [-1, 1] x [-1, 1] x [-1, 1] ou [0, 1] x [0, 1] x [0, 1].

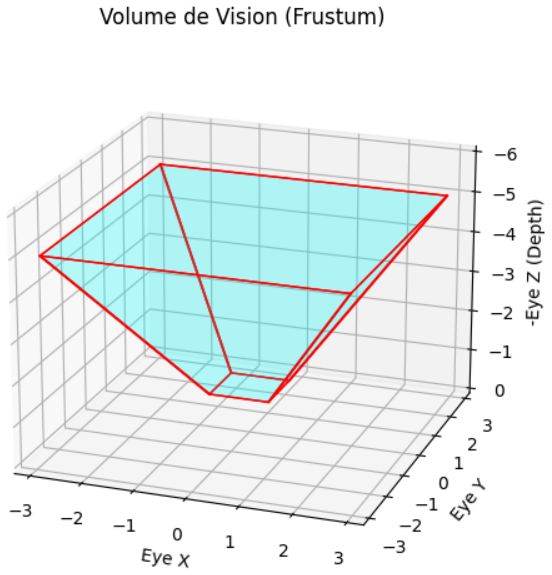


Figure 5: Clipping contre le volume de vision (frustum) dans l'espace caméra.

0.2.5 5. Transformation Écran (Projection)

- **Objectif :** Projeter les primitives 3D (après clipping et potentiellement normalisation en NDC) sur un plan image 2D (Screen Space).
- **Processus :** Applique une transformation de projection (perspective ou orthographique) qui transforme le volume de vision (frustum ou cube NDC) en coordonnées écran (pixels), tout en conservant généralement l'information de profondeur (z) pour l'étape de visibilité.

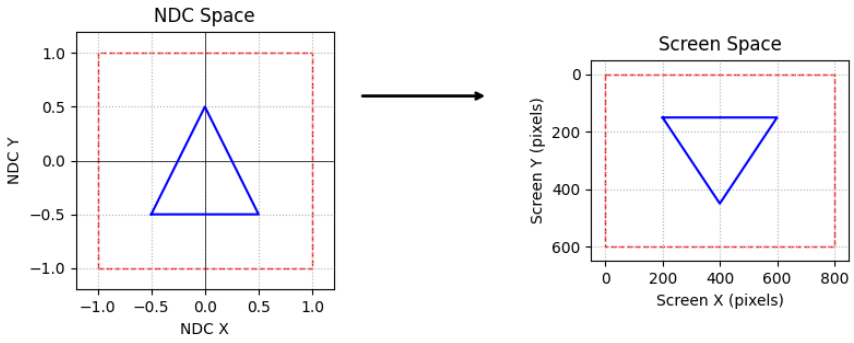


Figure 6: Passage des Coordonnées Normalisées (NDC) à l'espace écran (Screen Space).

0.2.6 6. Pixelisation (Rasterization)

- **Objectif :** Convertir les primitives 2D (définies par leurs sommets en coordonnées écran) en une série de fragments (pixels potentiels) qui couvrent la forme de la primitive.
- **Processus :** Pour chaque primitive (ex: triangle), détermine quels pixels de la grille de l'écran sont couverts par celle-ci. Interpole les attributs définis aux sommets (couleur, coordonnées de texture, profondeur) pour chaque fragment généré.

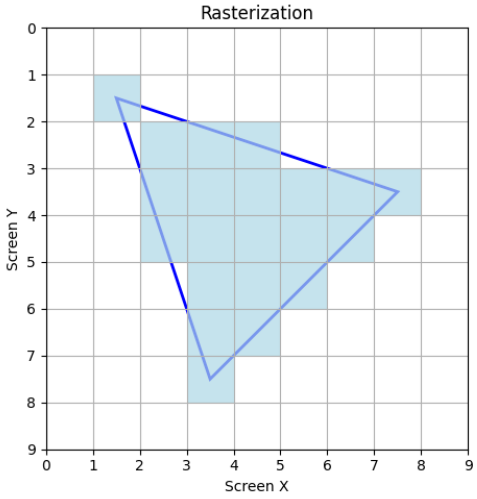


Figure 7: Découpage d'une primitive (triangle) en fragments (pixels).

0.2.7 7. Visibilité / Rendu

- **Objectif :** Déterminer, pour chaque pixel, quel fragment est réellement visible (gestion des recouvrements) et écrire la couleur finale dans le framebuffer (mémoire image).
- **Processus :**
  - **Élimination des parties cachées :** Souvent réalisée à l'aide d'un Z-buffer (tampon de profondeur) qui stocke la profondeur du fragment le plus proche pour chaque pixel. Un nouveau fragment n'est dessiné que s'il est plus proche que celui déjà stocké.

– **Remplissage du framebuffer :** La couleur du fragment visible (calculée lors de l'illumination et interpolée lors de la rastérisation) est écrite dans le pixel correspondant du framebuffer. D'autres opérations peuvent avoir lieu ici (mélange alpha pour la transparence, etc.).

0.2.8 Résumé des Systèmes de Coordonnées

Le passage à travers le pipeline implique plusieurs changements de systèmes de coordonnées :

1. **Repère Objet (Object Space)** : Coordonnées locales à chaque objet.
2. **Repère Scène (World Space)** : Espace commun où tous les objets sont placés.
3. **Repère Caméra (Eye Space)** : La scène vue depuis la caméra.
4. **Repère Caméra Normalisé (NDC)** : Coordonnées normalisées après projection et clipping, souvent dans un cube.
5. **Espace Écran (Screen Space)** : Coordonnées en pixels sur l'image finale.

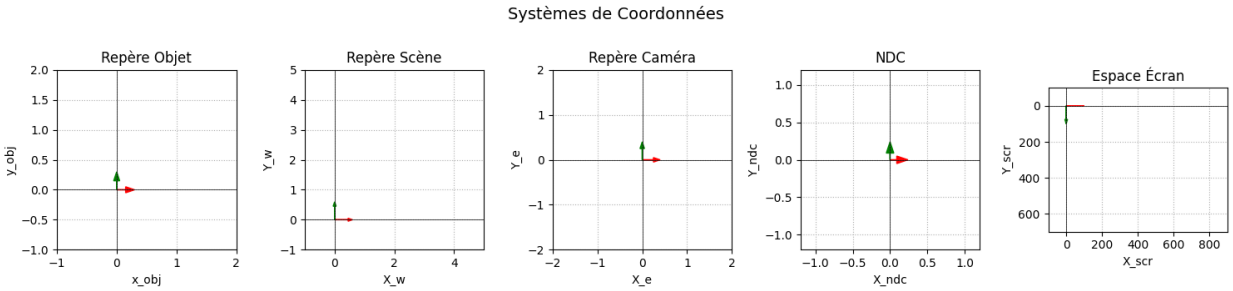


Figure 8: Illustration des différents systèmes de coordonnées.

0.2.9 Détails du Pipeline

Le pipeline traite séquentiellement chaque primitive (triangle).

- **Entrées :** Modèles géométriques (objets, surfaces), sources de lumière, modèle d'illumination, caméra (point de vue, frustum), fenêtre d'affichage (viewport), et format de sortie (couleurs, intensités, ex: 24 bits RVB).
- **Implémentation :** Peut être réalisé de diverses manières, combinant matériel (hardware) et logiciel (software). Certaines étapes peuvent offrir des points de programmation (ex: vertex shaders, pixel/fragment shaders) pour personnaliser le rendu.

0.3 II. Fondements Mathématiques

L'infographie repose fortement sur des concepts mathématiques pour décrire et manipuler les objets et la lumière dans l'espace.

0.3.1 Pourquoi des Mathématiques ?

De nombreux concepts graphiques font appel aux mathématiques :

- Systèmes de coordonnées (cartésiens, etc.)
- Transformations (translation, rotation, échelle, projection)
- Rayonnement (Ray-casting, Ray-tracing)

- Conversion des couleurs (espaces colorimétriques)
- Tests d'intersection (objets, rayons)
- Requêtes géométriques (distances, angles)
- Simulations physiques (gravité, collisions)
- Et bien d'autres...

Les deux piliers mathématiques principaux sont l'algèbre linéaire et l'analyse vectorielle.

0.3.2 Algèbre Linéaire

Matrices :

Definition 0.2. Une matrice M est un tableau rectangulaire (à deux indices) de $n \times m$ valeurs numériques extraites d'un même ensemble $\mathbb{E}$ (par exemple, $\mathbb{R}$).

$$M = \begin{pmatrix} m_{11} & m_{12} & \cdots & m_{1m} \\ m_{21} & m_{22} & \cdots & m_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nm} \end{pmatrix}$$

Où n est le nombre de lignes et m le nombre de colonnes. Si $n = m$, la matrice est dite **carrée**.

Exemple ($n = 4, m = 5$):

$$M = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} & m_{15} \\ m_{21} & m_{22} & m_{23} & m_{24} & m_{25} \\ m_{31} & m_{32} & m_{33} & m_{34} & m_{35} \\ m_{41} & m_{42} & m_{43} & m_{44} & m_{45} \end{pmatrix}$$

Opérations possibles :

- **Addition :**
  - Matrice-matrice (si mêmes dimensions) : $(A + B)_{ij} = A_{ij} + B_{ij}$
  - Scalaire-matrice : $(s + M)_{ij} = s + M_{ij}$ (moins courant) ou $(sM)_{ij} = s \times M_{ij}$ (multiplication scalaire)
- **Multiplication :**
  - Scalaire-matrice : $(sM)_{ij} = s \times M_{ij}$
  - Matrice-vecteur : voir ci-dessous.
  - Matrice-matrice : voir ci-dessous.
- **Inversion :** Pour les matrices carrées inversibles M^{-1} tel que $MM^{-1} = M^{-1}M = I$ (identité).
- **Transposition :** $(M^T)_{ij} = M_{ji}$ (échange lignes et colonnes).

Produit Matrice par Vecteur : Soient une matrice M de dimension $m \times n$ et un vecteur colonne V de dimension n ($n \times 1$). Le vecteur résultant W est de dimension m ($m \times 1$) et est calculé comme suit : $W = M \cdot V$. Chaque composante w_i de W est le produit scalaire de la i-ème ligne de M avec le vecteur V :

$$w_i = \sum_{k=1}^{n} M_{ik} V_k \quad (1 \leq i \leq m)$$

Visuellement :

$$\left(\begin{array}{cccc} \cdots & \cdots & \cdots & \\ \hline m_{i1} & m_{i2} & \cdots & m_{in} \\ \hline \cdots & \cdots & \cdots & \end{array}\right)_{m\times n} \times \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}_{n\times 1} = \begin{pmatrix} \vdots \\ w_i \\ \vdots \end{pmatrix}_{m\times 1}$$

Exemple :

$$\begin{pmatrix} 2 & 4 \\ 6 & 8 \\ 10 & 12 \end{pmatrix}_{3\times 2} \times \begin{pmatrix} 1 \\ 3 \end{pmatrix}_{2\times 1} = \begin{pmatrix} (2\times 1)+(4\times 3) \\ (6\times 1)+(8\times 3) \\ (10\times 1)+(12\times 3) \end{pmatrix} = \begin{pmatrix} 14 \\ 30 \\ 46 \end{pmatrix}_{3\times 1}$$

Produit Matrice par Matrice : Soient deux matrices M_1 de dimension $n\times m$ et M_2 de dimension $m\times p$. La matrice produit M est de dimension $n\times p$. Chaque élément m_{ij} de M est le produit scalaire de la i-ème ligne de M_1 avec la j-ème colonne de M_2 :

$$m_{ij} = \sum_{k=1}^{m} (M_1)_{ik}(M_2)_{kj} \quad (1 \le i \le n, 1 \le j \le p)$$

Visuellement :

$$\left(\begin{array}{c} \vdots \\ \hline (M_1)_{i1}\cdots(M_1)_{im} \\ \hline \vdots \end{array}\right)_{n\times m} \times \begin{pmatrix} \cdots & (M_2)_{1j} & \cdots \\ & \vdots & \\ \cdots & (M_2)_{mj} & \cdots \end{pmatrix}_{m\times p} = \begin{pmatrix} \ddots & \vdots & \\ \cdots & m_{ij} & \cdots \\ & \vdots & \ddots \end{pmatrix}_{n\times p}$$

Exemple :

$$\begin{pmatrix} 1 & 3 & 5 \\ 7 & 9 & 11 \end{pmatrix}_{2\times 3} \times \begin{pmatrix} 2 & 4 \\ 6 & 8 \\ 10 & 12 \end{pmatrix}_{3\times 2} = \begin{pmatrix} (1\cdot 2+3\cdot 6+5\cdot 10) & (1\cdot 4+3\cdot 8+5\cdot 12) \\ (7\cdot 2+9\cdot 6+11\cdot 10) & (7\cdot 4+9\cdot 8+11\cdot 12) \end{pmatrix} = \begin{pmatrix} 70 & 88 \\ 178 & 232 \end{pmatrix}_{2\times 2}$$

0.3.3 Analyse Vectorielle

Scalaires :

- Un scalaire est une grandeur totalement définie par un **nombre** et une unité (optionnelle en maths pures).
- Il a une valeur numérique mais **pas d'orientation**.
- Exemples : Masse, Distance, Température, Volume, Densité, etc.
- Les scalaires obéissent aux lois de l'algèbre ordinaire.
- **Opérations élémentaires :** Addition ($+$), Multiplication ($\cdot$).
- **Propriétés :**
  - Commutativité : $\alpha+\beta=\beta+\alpha$, $\alpha\cdot\beta=\beta\cdot\alpha$
  - Associativité : $\alpha+(\beta+\gamma)=(\alpha+\beta)+\gamma$, $\alpha\cdot(\beta\cdot\gamma)=(\alpha\cdot\beta)\cdot\gamma$
  - Distributivité : $\alpha\cdot(\beta+\gamma)=(\alpha\cdot\beta)+(\alpha\cdot\gamma)$
- **Identité :**
  - Addition : 0 ($\alpha+0=0+\alpha=\alpha$)
  - Multiplication : 1 ($\alpha\cdot 1=1\cdot\alpha=\alpha$)

Vecteurs :

- Un vecteur est une entité mathématique définie par n valeurs numériques (composantes) extraites du même ensemble $\mathbb{E}$ (e.g., $\mathbb{N}, \mathbb{Z}, \mathbb{R}, \mathbb{C}$).
- Ces valeurs décrivent le **module** (longueur) et l'**orientation** du vecteur.
- n est appelé la **dimension** du vecteur. On dit que le vecteur est défini dans E^n. E^n est un espace vectoriel de dimension n.
- **Exemple :** En $\mathbb{R}^3$, un vecteur $\vec{v}$ peut être écrit comme $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$.
- **Usages :** Déplacement, Vitesse, Accélération, Force.
- Les vecteurs obéissent aux lois de l'algèbre vectorielle.
- **Opérations élémentaires :** Addition de vecteurs, multiplication par un scalaire, produit scalaire, produit vectoriel (en dim 3), normalisation.

Vecteurs Unitaires et Base :

- Dans un repère, les vecteurs sont décrits dans une **base**, qui est un ensemble de vecteurs linéairement indépendants (unités de l'ensemble).
- La base est généralement **orthonormée** (vecteurs unitaires et perpendiculaires entre eux). En 3D, on utilise souvent la base canonique $(\vec{i}, \vec{j}, \vec{k})$.
- Un vecteur $\vec{V}$ est représenté par l'addition des vecteurs de base, multipliés chacun par la **projection** (composante) de $\vec{V}$ sur l'axe correspondant.

$$\vec{V} = x'\vec{i} + y'\vec{j} + z'\vec{k}$$

où x', y', z' sont les composantes de $\vec{V}$ dans la base $(\vec{i}, \vec{j}, \vec{k})$.

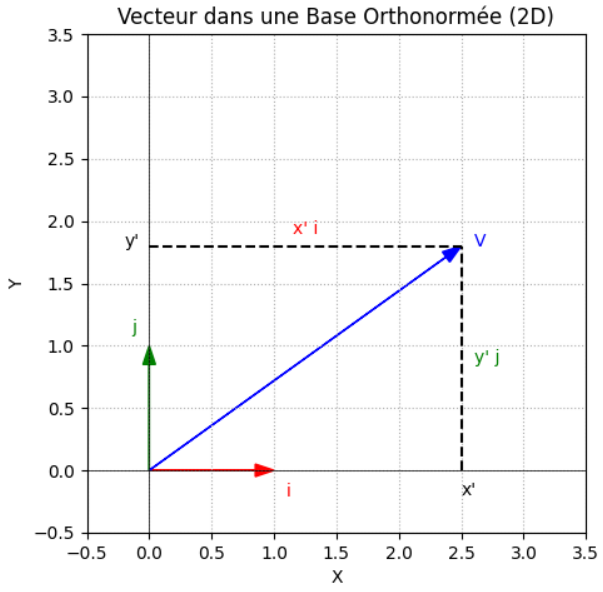


Figure 9: Représentation d'un vecteur par ses composantes dans une base.

Addition de Vecteurs :

- **Règle du parallélogramme :** Géométriquement, la somme $\vec{A}+\vec{B}$ est la diagonale du parallélogramme formé par $\vec{A}$ et $\vec{B}$.
- **Addition des composantes :** Algébriquement, on additionne les composantes correspondantes : Si $\vec{A} = a_x\vec{i} + a_y\vec{j} + a_z\vec{k}$ et $\vec{B} = b_x\vec{i} + b_y\vec{j} + b_z\vec{k}$, alors

$$\vec{A} + \vec{B} = (a_x + b_x)\vec{i} + (a_y + b_y)\vec{j} + (a_z + b_z)\vec{k}$$

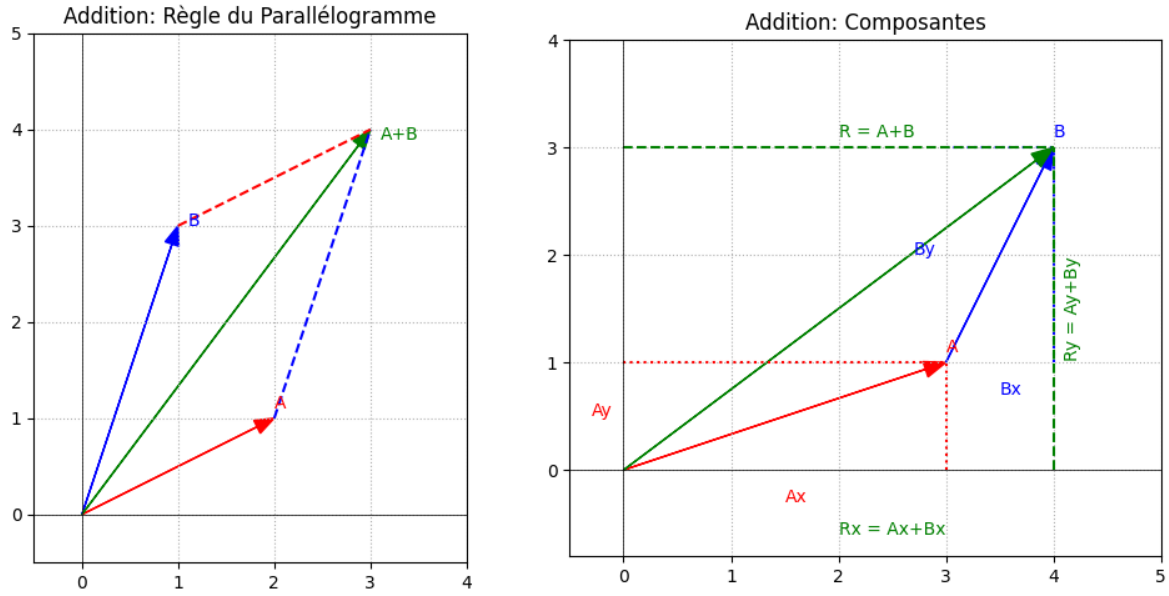


Figure 10: Addition de vecteurs : règle du parallélogramme et addition des composantes.

Norme d'un Vecteur :

- La norme (ou module) d'un vecteur $\vec{V}$, notée $||\vec{V}||$ ou $|\vec{V}|$, est sa longueur.
- Elle est calculée par le théorème de Pythagore en utilisant ses composantes.
- En 2D : $||\vec{V}|| = ||x'\vec{i} + y'\vec{j}|| = \sqrt{(x')^2 + (y')^2}$
- En 3D : $||\vec{V}|| = ||x'\vec{i} + y'\vec{j} + z'\vec{k}|| = \sqrt{(x')^2 + (y')^2 + (z')^2}$
- En dimension n, on généralise : la norme est la racine carrée de la somme des carrés des composantes.
- La norme du vecteur $\vec{AB}$ (reliant le point A au point B) est la distance entre A et B.

Normalisation d'un Vecteur :

- La normalisation transforme un vecteur non nul en un **vecteur unitaire** (de norme 1) ayant la même direction et le même sens.
- Pour normaliser un vecteur $\vec{V}$, il suffit de diviser chacune de ses composantes par sa norme $||\vec{V}||$.

- Le vecteur normalisé $\hat{V}$ est : $\hat{V} = \frac{\vec{V}}{||\vec{V}||} = \frac{1}{||\vec{V}||}(x'\vec{i} + y'\vec{j} + z'\vec{k})$

$$\hat{V} = \left(\frac{x'}{||\vec{V}||}, \frac{y'}{||\vec{V}||}, \frac{z'}{||\vec{V}||} \right)$$

où $||\vec{V}|| = \sqrt{(x')^2 + (y')^2 + (z')^2}$

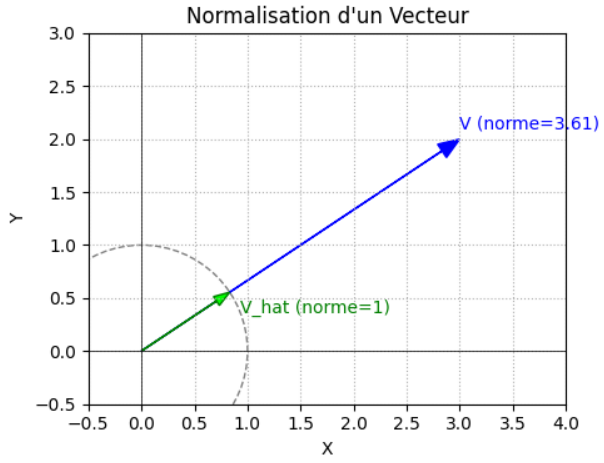


Figure 11: Un vecteur $\vec{V}$ et son vecteur normalisé $\hat{V}$.

Produit Scalaire (Dot Product) :

- Le produit scalaire de deux vecteurs $\vec{A}$ et $\vec{B}$ est un **scalaire**.
- **Définition géométrique :** $\vec{A} \cdot \vec{B} = ||\vec{A}||\,||\vec{B}|| \cos\theta$, où θ est l'angle entre les deux vecteurs. Cela correspond au produit du module de l'un par la projection du second sur le premier.
- **Définition par composantes :** $\vec{A} \cdot \vec{B} = A_x B_x + A_y B_y + A_z B_z$.
- **Propriétés :**
  - Commutativité : $\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u}$
  - Distributivité par l'addition : $(\vec{u} + \vec{v}) \cdot \vec{w} = \vec{u} \cdot \vec{w} + \vec{v} \cdot \vec{w}$
  - Distributivité par un scalaire : $\vec{u} \cdot (k\vec{v}) = k(\vec{u} \cdot \vec{v})$
- **Angle entre deux vecteurs :** On peut déduire l'angle θ :

$$\cos\theta = \frac{\vec{A} \cdot \vec{B}}{||\vec{A}||\,||\vec{B}||} = \frac{A_x B_x + A_y B_y + A_z B_z}{\sqrt{A_x^2 + A_y^2 + A_z^2}\sqrt{B_x^2 + B_y^2 + B_z^2}}$$

- **Signe et Orthogonalité :**
  - $\vec{A} \cdot \vec{B} > 0 \implies -90° < \theta < 90°$ (angle aigu)
  - $\vec{A} \cdot \vec{B} = 0 \implies \theta = \pm 90°$ (vecteurs orthogonaux)

- $\vec{A} \cdot \vec{B} < 0 \implies 90° < \theta < 180°$ ou $-180° < \theta < -90°$ (angle obtus)

- **Applications :**
  - Projection d'un vecteur sur un autre.
  - Élimination des faces cachées (back-face culling) : si le produit scalaire entre la normale à la face et la direction de vue est positif, la face est tournée vers l'arrière.
  - Calcul d'angle entre vecteurs.
  - Calcul de la quantité de lumière perçue (lois de Lambert, Phong).
  - Ombrage.

Produit Vectoriel (Cross Product) : (Principalement en 3D)

- Le produit vectoriel de deux vecteurs $\vec{A}$ et $\vec{B}$ (noté $\vec{A} \times \vec{B}$) est un **vecteur**.
- **Direction :** $\vec{A} \times \vec{B}$ est perpendiculaire au plan formé par $\vec{A}$ et $\vec{B}$. Son sens est donné par la règle de la main droite (si on tourne $\vec{A}$ vers $\vec{B}$, le pouce indique la direction de $\vec{A} \times \vec{B}$).
- **Module :** $||\vec{A} \times \vec{B}|| = ||\vec{A}||\,||\vec{B}||\,|\sin\theta|$, où θ est l'angle entre $\vec{A}$ et $\vec{B}$. Le module correspond à l'aire du parallélogramme formé par $\vec{A}$ et $\vec{B}$.
- Le produit vectoriel est nul si les vecteurs sont parallèles ($\theta = 0°$ or $180°$) et maximal s'ils sont perpendiculaires ($\theta = 90°$).
- **Calcul par composantes :**

$$\vec{A} \times \vec{B} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ A_x & A_y & A_z \\ B_x & B_y & B_z \end{vmatrix} = (A_y B_z - A_z B_y)\vec{i} - (A_x B_z - A_z B_x)\vec{j} + (A_x B_y - A_y B_x)\vec{k}$$

  Exemple : $\vec{A} = (1, 2, -4)$, $\vec{B} = (3, -1, 5)$

$$\begin{aligned} \vec{A} \times \vec{B} &= \vec{i}(2 \times 5 - (-4) \times (-1)) - \vec{j}(1 \times 5 - (-4) \times 3) + \vec{k}(1 \times (-1) - 2 \times 3) \\ &= \vec{i}(10 - 4) - \vec{j}(5 + 12) + \vec{k}(-1 - 6) \\ &= 6\vec{i} - 17\vec{j} - 7\vec{k} \end{aligned}$$

- **Propriétés :**
  - Anticommutativité : $\vec{u} \times \vec{v} = -(\vec{v} \times \vec{u})$
  - Distributivité sur l'addition : $(\vec{u} + \vec{v}) \times \vec{w} = \vec{u} \times \vec{w} + \vec{v} \times \vec{w}$
  - Distributivité par un scalaire : $(k\vec{u}) \times \vec{v} = \vec{u} \times (k\vec{v}) = k(\vec{u} \times \vec{v})$
  - Non-associativité : $(\vec{u} \times \vec{v}) \times \vec{w} \neq \vec{u} \times (\vec{v} \times \vec{w})$ (en général)
- **Application :** Calcul de la **normale** à un plan (ou une face de triangle). Si $P1\vec{P}2$ et $P1\vec{P}3$ sont deux vecteurs arêtes d'un triangle, $P1\vec{P}2 \times P1\vec{P}3$ donne un vecteur normal à la face.

0.4 III. Implémentation du Pipeline Graphique

Le pipeline graphique peut être implémenté de différentes manières, allant du tout logiciel aux solutions matérielles dédiées.

0.4.1 Implémentation Logicielle (Software Configurable)

- Toutes les étapes du pipeline sont exécutées par le CPU (processeur central).
- Configuration : Entièrement logicielle.
- Performance : Limitée, surtout pour les scènes complexes et le rendu en temps réel.
- Historique : Approche initiale avant l'avènement des cartes graphiques dédiées.

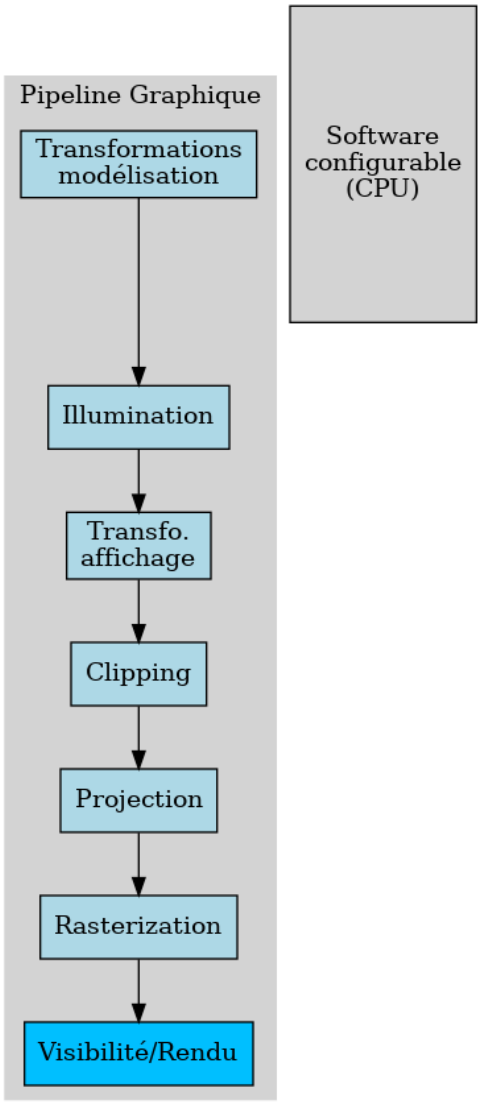


Figure 12: Implémentation entièrement logicielle (CPU).

0.4.2 Premières Cartes Graphiques (Hardware)

- Certaines étapes clés (notamment la rastérisation) sont accélérées par du matériel dédié (Hardware).
- Configuration : Le logiciel configure le matériel.
- Performance : Amélioration significative par rapport au tout logiciel.

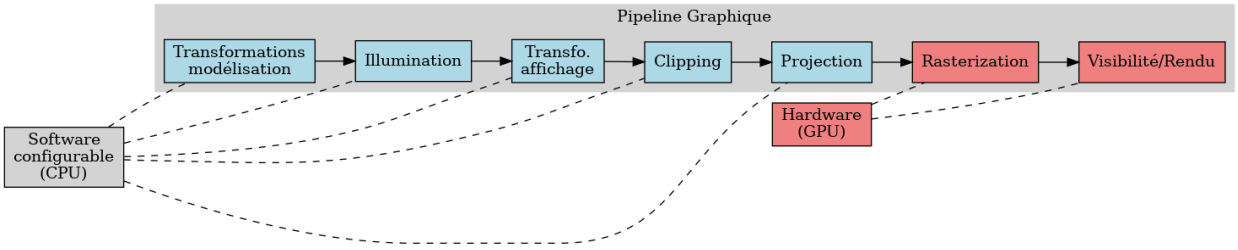


Figure 13: Implémentation avec accélération matérielle (Hardware) pour certaines étapes.

0.4.3 Cartes Graphiques Configurables (Hardware Configurable)

- Davantage d'étapes sont implémentées en matériel.
- Le matériel offre des options de configuration, mais les opérations restent largement fixes (pipeline à fonction fixe).
- Performance : Encore améliorée.

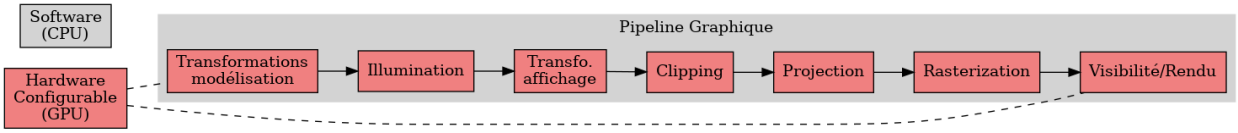


Figure 14: Implémentation avec matériel configurable pour la plupart des étapes.

0.4.4 Cartes Graphiques Programmables (Hardware Programmable)

- C'est l'architecture moderne des GPU (Graphics Processing Units).
- Certaines étapes clés (typiquement liées aux sommets/vertices et aux fragments/pixels) deviennent programmables via des **shaders** (petits programmes exécutés sur le GPU).
- Offre une flexibilité énorme pour créer des effets visuels personnalisés.
- Le matériel est hautement parallèle pour traiter des millions de sommets et de fragments simultanément.

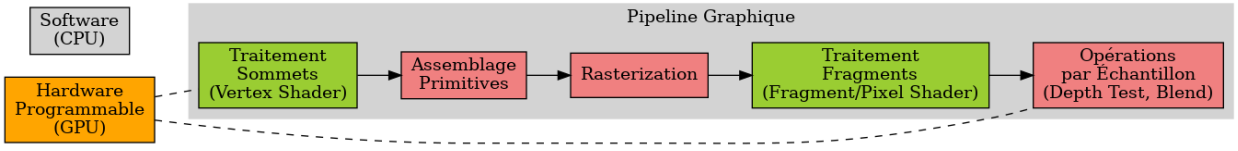


Figure 15: Implémentation avec matériel programmable (GPU moderne).

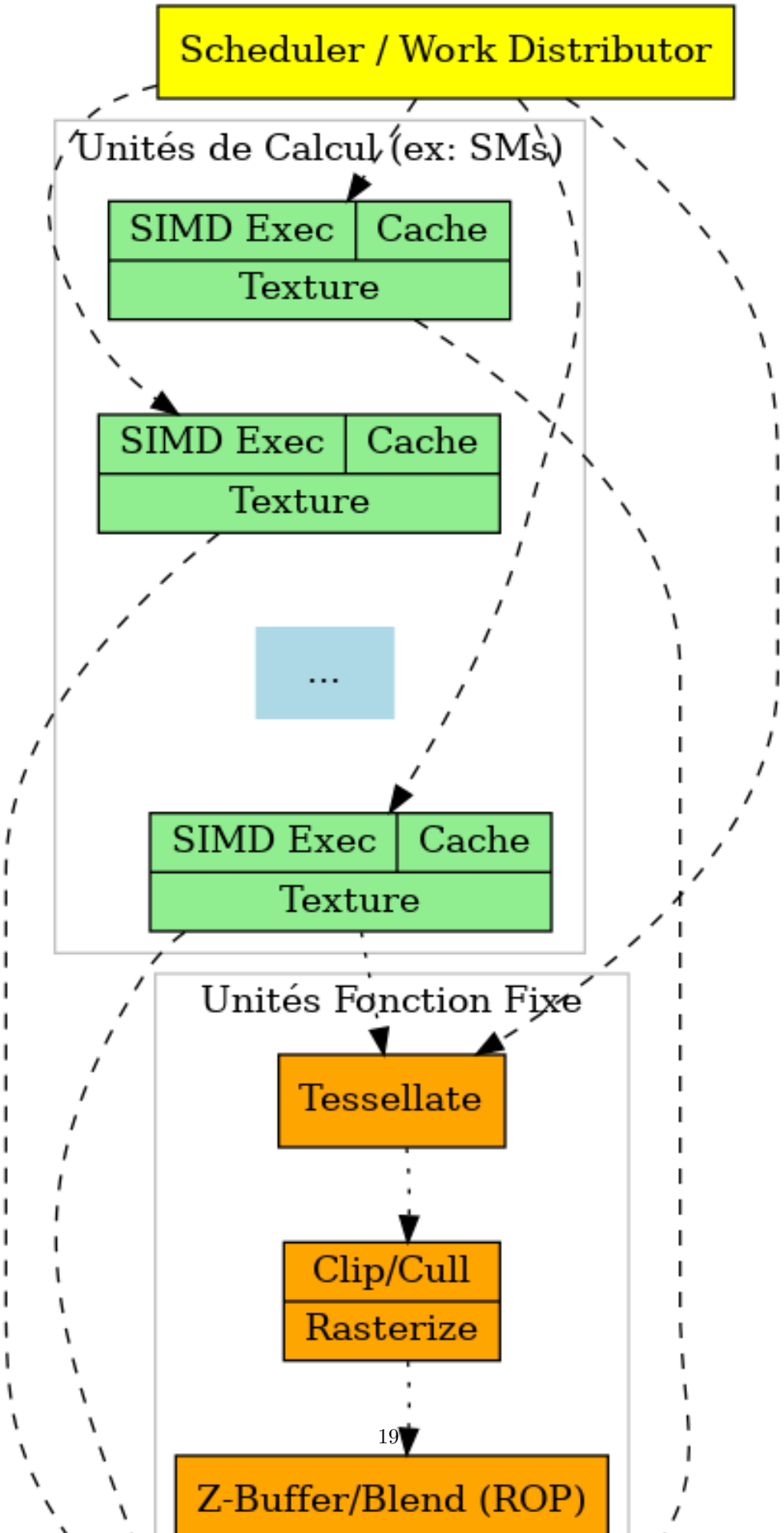
0.4.5 Le GPU (Graphics Processing Unit)

- **Matériel spécialisé :** Conçu pour les calculs massivement parallèles typiques de l'infographie. Très bon dans l'exécution de quelques opérations simples sur d'énormes volumes de données (SIMD/SIMT - Single Instruction, Multiple Data/Thread).
- **Architecture :** Composé de nombreux cœurs de calcul simples regroupés en unités d'exécution, avec des hiérarchies de caches et un accès rapide à une mémoire dédiée (VRAM).

  - Unités d'exécution (SIMD/Streaming Multiprocessors).
  - Caches (L1, L2).
  - Unités spécialisées (Texture mapping units, Raster Operations Pipelines - ROPs, Tessellation units, Ray Tracing cores - sur les GPU récents).
  - Scheduler / Work Distributor : Gère la distribution des tâches (sommets, fragments) aux unités d'exécution.
  - Interface Mémoire : Connexion à la mémoire GPU (GDDR).

- **Performance :** Peut exécuter la même opération (shader) sur des millions de points/pixels de données à la fois. Idéal lorsque le même code est exécuté sur de nombreuses données indépendantes. Moins efficace si le code diverge fortement entre les threads (mauvais pour les branchements conditionnels complexes).

- **Catalogue d'opérations réduit :** Optimisé pour les opérations mathématiques (algèbre linéaire, interpolations) courantes en graphisme.


Scheduler / Work Distributor
Unités de Calcul (ex: SMs)
SIMD Exec
Cache
Texture
SIMD Exec
Cache
Texture
...
SIMD Exec
Cache
Texture
Unités Fonction Fixe
Tessellate
Clip/Cull
Rasterize
Z-Buffer/Blend (ROP)

0.5 IV. Espaces Vectoriels et Affines

Ces concepts mathématiques sont fondamentaux pour structurer l'espace dans lequel évoluent les objets graphiques.

- **Espace Vectoriel :** C'est l'espace où vivent les **vecteurs** (déplacements, directions). Ses propriétés principales incluent l'existence d'un vecteur nul ($\vec{0}$) et la stabilité par combinaison linéaire (si $\vec{u}$ et $\vec{v}$ sont dans l'espace, alors $\alpha\vec{u} + \beta\vec{v}$ l'est aussi, pour tous scalaires α, β). Un espace vectoriel n'a pas d'origine "privilégiée".
- **Espace Affine :** C'est l'espace où vivent les **points** et où l'on définit les objets géométriques usuels (droites, plans, etc.). Il se construit à partir :
  - D'un point de référence (l'**origine** O).
  - D'un **espace vectoriel** associé (l'ensemble des déplacements autorisés à partir de n'importe quel point).

  Un point P peut être vu comme la somme de l'origine O et d'un vecteur $\vec{OP}$ appartenant à l'espace vectoriel associé. La différence entre deux points est un vecteur : $\vec{AB} = B - A$. L'addition d'un point et d'un vecteur est un point : $A + \vec{v} = B$. L'addition de deux points n'est généralement pas définie directement dans ce cadre (mais le barycentre l'est).

0.6 Conclusion

Ce chapitre a introduit les concepts fondamentaux de l'infographie, de son histoire et ses applications à son mécanisme central, le pipeline graphique. Nous avons vu que chaque étape du pipeline transforme les données géométriques et d'apparence pour aboutir à l'image finale. L'implémentation de ce pipeline a évolué de solutions purement logicielles vers des architectures matérielles massivement parallèles et programmables (GPU).

Au cœur de ces processus se trouvent les mathématiques : l'algèbre linéaire (matrices pour les transformations) et l'analyse vectorielle (vecteurs pour les positions, directions, normales ; produits scalaires et vectoriels pour les calculs d'angles, de projections, d'orientation). La maîtrise de ces outils mathématiques et la compréhension du pipeline graphique sont essentielles pour quiconque s'intéresse à la création ou à l'analyse d'images de synthèse, y compris dans le contexte de la science des données où la visualisation joue un rôle crucial.