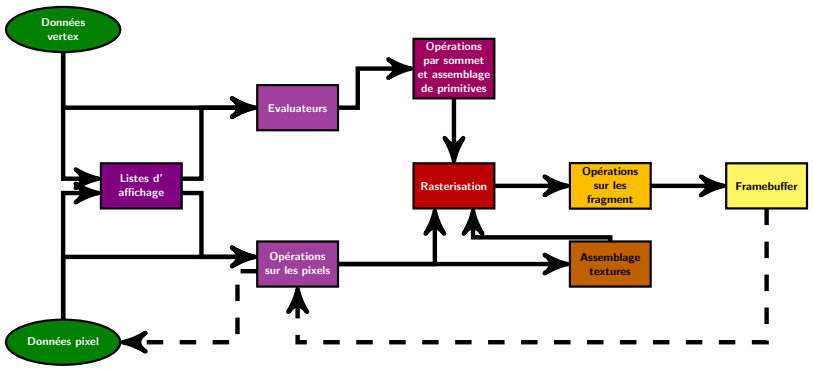
Anatomie des cartes graphiques accélératrices 3D

ENSIIE2 : Option RVIG

Jean-Yves Didier

Université d'Evry-val d'Essonne



Carte graphique et accélération 3D

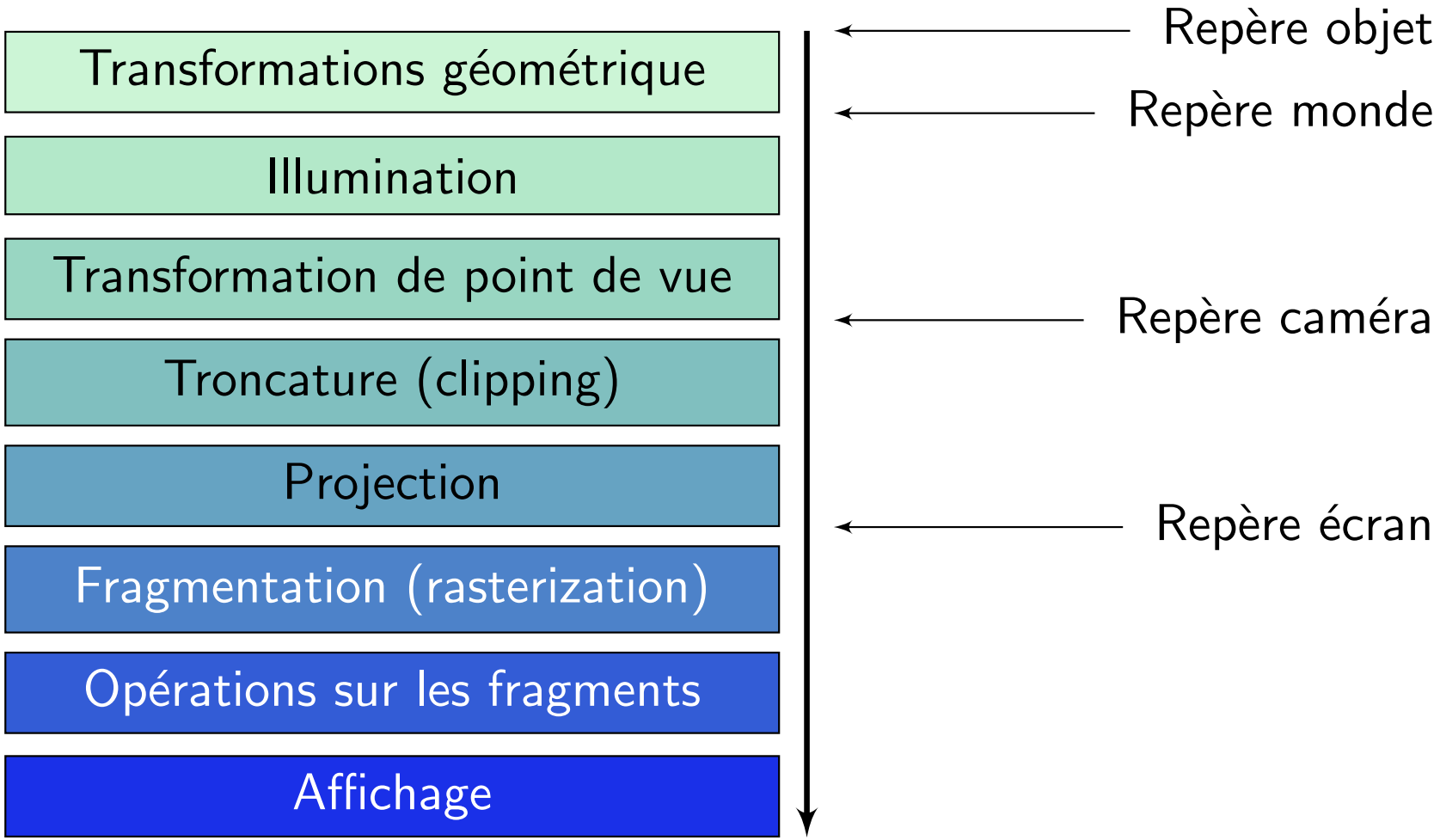
Carte graphique

Adaptateur générant une image puis un signal compatible avec l'écran.

Accélération 3D

- Est apparue au début des années 1990 ;
- En entrée : un ensemble de primitives 3D :
  - Sommets (coordonnées 3D), segments, polygones, textures.
- En sortie : des pixels ;
- Fonction : transformer des données 3D en images 2D :
  - Notion de *pipeline* graphique.

Le pipeline graphique : point de vue fonctionnel



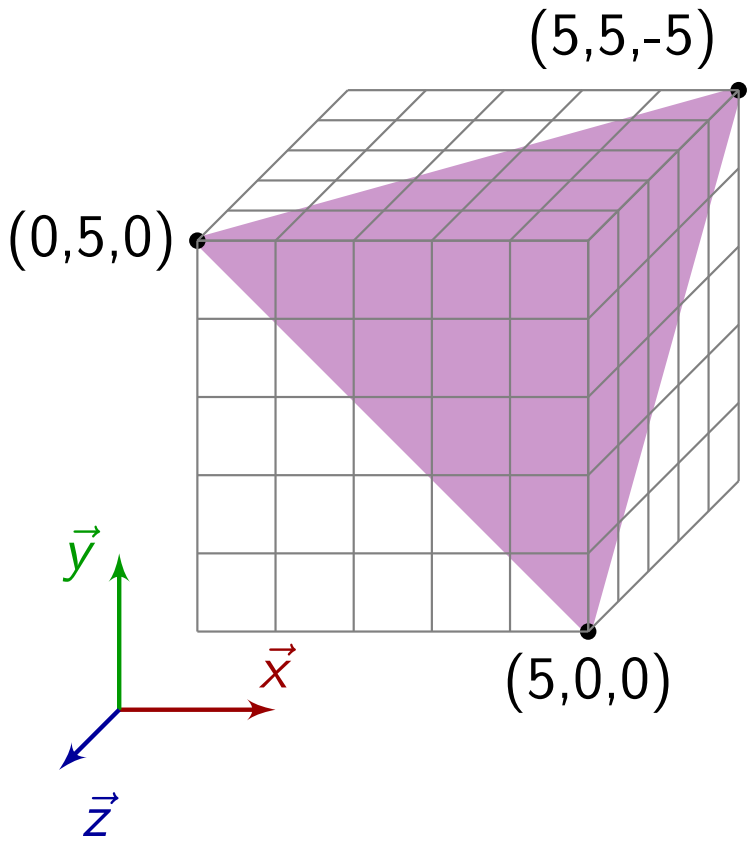
Opération sur les fragments

Scène 3D = polygones + textures.

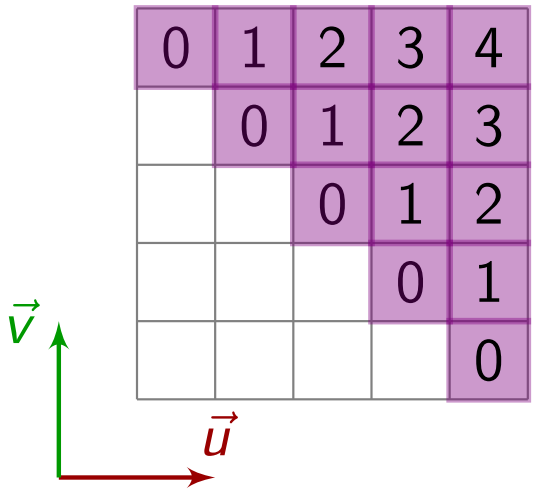
Le travail de la carte graphique

1. Décomposer polygones et textures en fragments (*rasterization*) ;
   - Propriétés des fragments :
     - 1 fragment = 1 pixel dans l'image finale ;
     - Paramètres associés : u, v, profondeur, couleur.
2. Traiter les fragments :
   - Les fragments passent au travers de différents tamis (*buffer*) ;
     - Les tamis sont programmables ;
     - Taille du tamis = taille de l'image finale.

Fragmentation



Espace caméra



Espace écran

Calcul des profondeurs

Profondeur = distance du fragment par rapport à la caméra

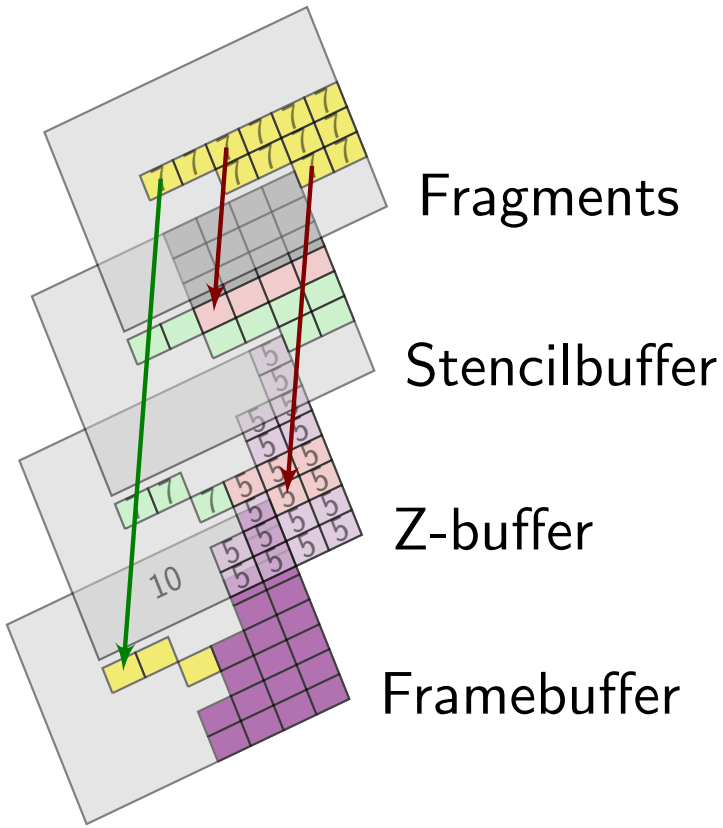
Opérations sur les fragments

Les tamis (*buffers*) usuels

Framebuffer : contient l'image finale ;

Z-buffer : test de profondeur ;

Stencilbuffer : masque ou pochoir.



Le pipeline graphique

A l'origine

Majoritairement logiciel :

- Exécution sur le CPU (*Central Processing Unit*) ;
- Processeur non spécialisé.

Maintenant

Presque entièrement matériel :

- Exécution sur le GPU (G = *Graphical*) ;
- Processeur spécialisé ;
- Gain en vitesse d'exécution.

API graphiques 3D

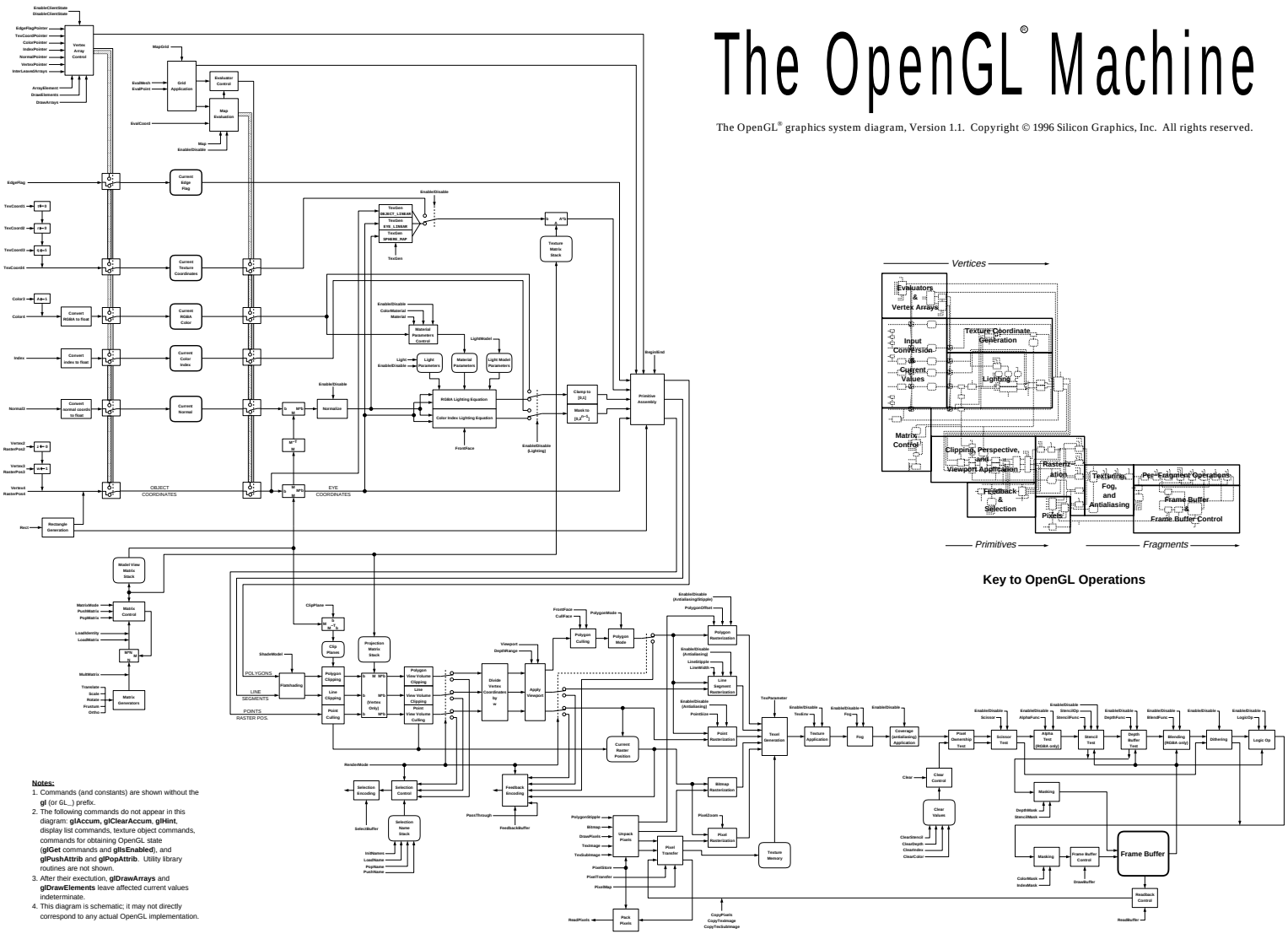
Des APIs de programmation

- *API = Application Programming Interface* ;
- Des interfaces logicielles (bibliothèques) pour des composants matériels dédiés au rendu graphique ;
- Derniers standards en date :
  - OpenGL 4.5 (août 2014) ;
  - Vulkan 1.0 (février 2016) ;
  - Direct3D 12 (maj août 2016).

Des interfaces logicielles de bas niveau

- Conçues pour piloter le matériel directement ;
- Paramètrent un automate qui gère le pipeline graphique ;
- Chaque mise à jour majeure d'une API correspond à une nouvelle génération de carte graphique.

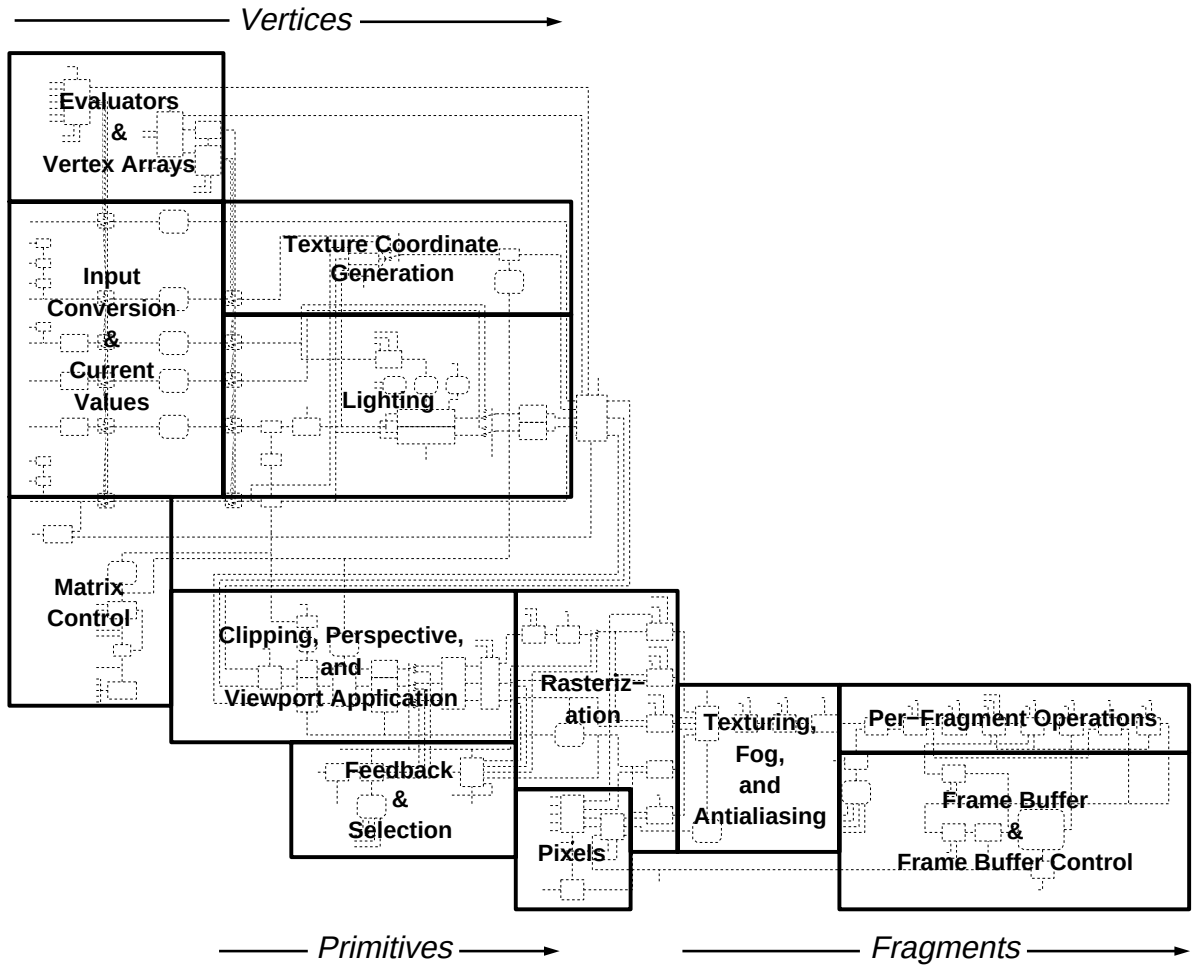
The OpenGL® Machine



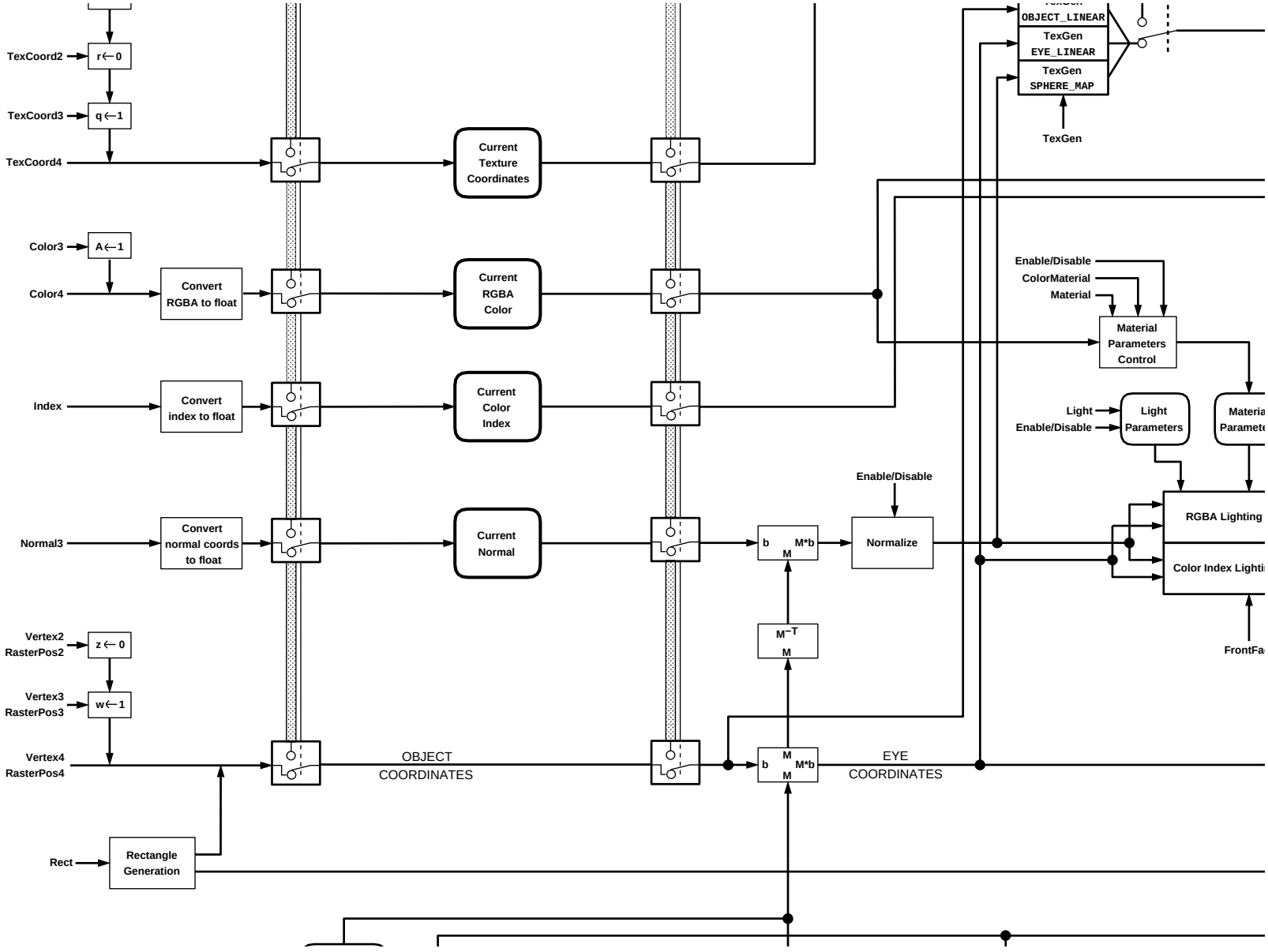
Key to OpenGL Operations

Notes:

1. Commands (and constants) are shown without the **gl** (or GL_) prefix.
2. The following commands do not appear in this diagram: **glAccum, glClearAccum**, **glHint**, display list commands, texture object commands, commands for obtaining OpenGL state (**glGet** commands and **glIsEnabled**), and **glPushAttrib** and **glPopAttrib**. Utility library routines are not shown.
3. After their exectution, **glDrawArrays** and **glDrawElements** leave affected current values indeterminate.
4. This diagram is schematic; it may not directly correspond to any actual OpenGL implementation.



Key to OpenGL Operations


TexCoord2
r←0
TexCoord3
q←1
TexCoord4
Current Texture Coordinates
OBJECT_LINEAR
TexGen EYE_LINEAR
TexGen SPHERE_MAP
TexGen
Color3
A←1
Color4
Convert RGBA to float
Current RGBA Color
Enable/Disable
ColorMaterial
Material
Material Parameters Control
Index
Convert index to float
Current Color Index
Light
Enable/Disable
Light Parameters
Enable/Disable
Normal3
Convert normal coords to float
Current Normal
b M*b M
Normalize
RGBA Lighting
Color Index Lighti
FrontFa
Vertex2 RasterPos2
z←0
Vertex3 RasterPos3
w←1
M−T M
Vertex4 RasterPos4
OBJECT COORDINATES
b M M*b M
EYE COORDINATES
Rect
Rectangle Generation

OpenGL (1/2)

Le premier standard

- Créé en 1992 par Silicon Graphics Incorporated (SGI) ;
- GL = *Graphics Library* ;
- Actuellement mis à jour par le Khronos Group (2006) :
  - consortium d'universités, de fabricants de cartes graphiques et d'éditeurs logiciels ;
  - membres connus : NVidia, Samsung, Nokia, Valve, Sony, Epic Games, Intel, Google, Qualcomm, AMD, Apple, ...

OpenGL (2/2)

Statut actuel

- Transition vers l'API Vulkan ;
- Multi-plateforme ;
- Standard ouvert ;
- A néanmoins un temps de retard sur Direct3D.

Un écosystème

- OpenGL : bibliothèque de programmation 3D en C ;
- OpenGL ES : variante pour les systèmes embarqués ;
- Vulkan : successeur d'OpenGL 4 et d'OpenGL ES 3 ;
- WebGL : version javascript pour le Web ;
- GLSL : langage de shader.

Direct3D

L'émancipation de Microsoft

- 1996, Microsoft sort DirectX, une collection de bibliothèques de fonction multimédias comprenant Direct3D ;
- 1999, Microsoft quitte le consortium « OpenGL »et se concentre sur DirectX ;
- Pendant un temps, une fusion est envisagée entre OpenGL et DirectX sous le nom de projet Fahrenheit (abandonné en 2000).

Caractéristiques de Direct3D

- Produit développé pour Windows ;
- Standard propriétaire ;
- Marque depuis quelques années son avance sur OpenGL.

Evolution des API

Les grandes générations

I Pipeline configurable et *Transform and lighting* :
- OpenGL 1 (1992), Direct3D 7 (1999).

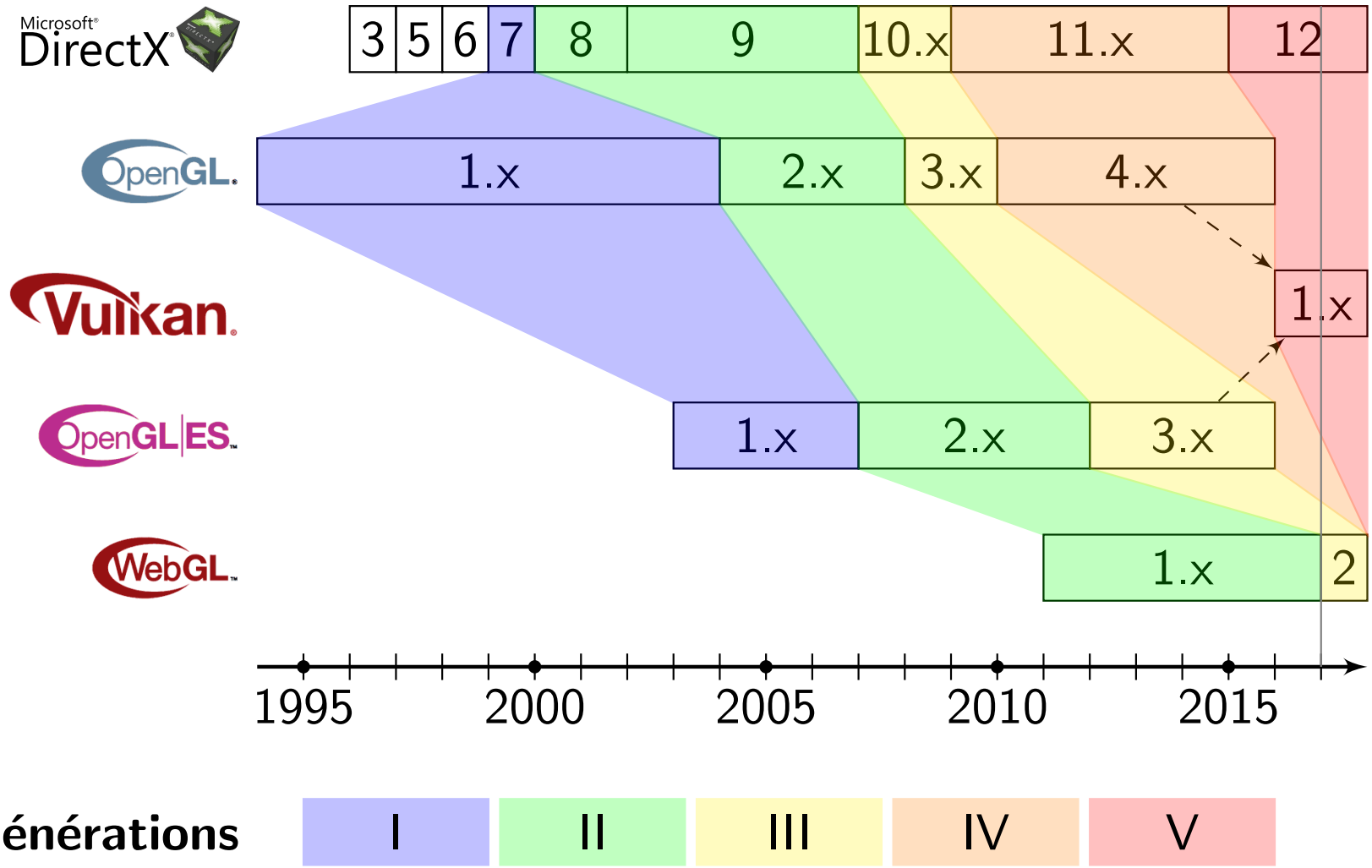
II Transition vers les pipelines programmables, *Vertex* et *fragments shaders* :
- OpenGL 2 (2004), Direct3D 8 (2000).

III *Geometry shaders* :
- OpenGL 3 (2008), Direct3D 10 (2007).

IV Tesselation et *compute shaders* :
- OpenGL 4 (2010), Direct3D 11 (2009).

V Optimisation du parallélisme bas niveau :
- Vulkan 1 (2016), Direct3D 12 (2015).

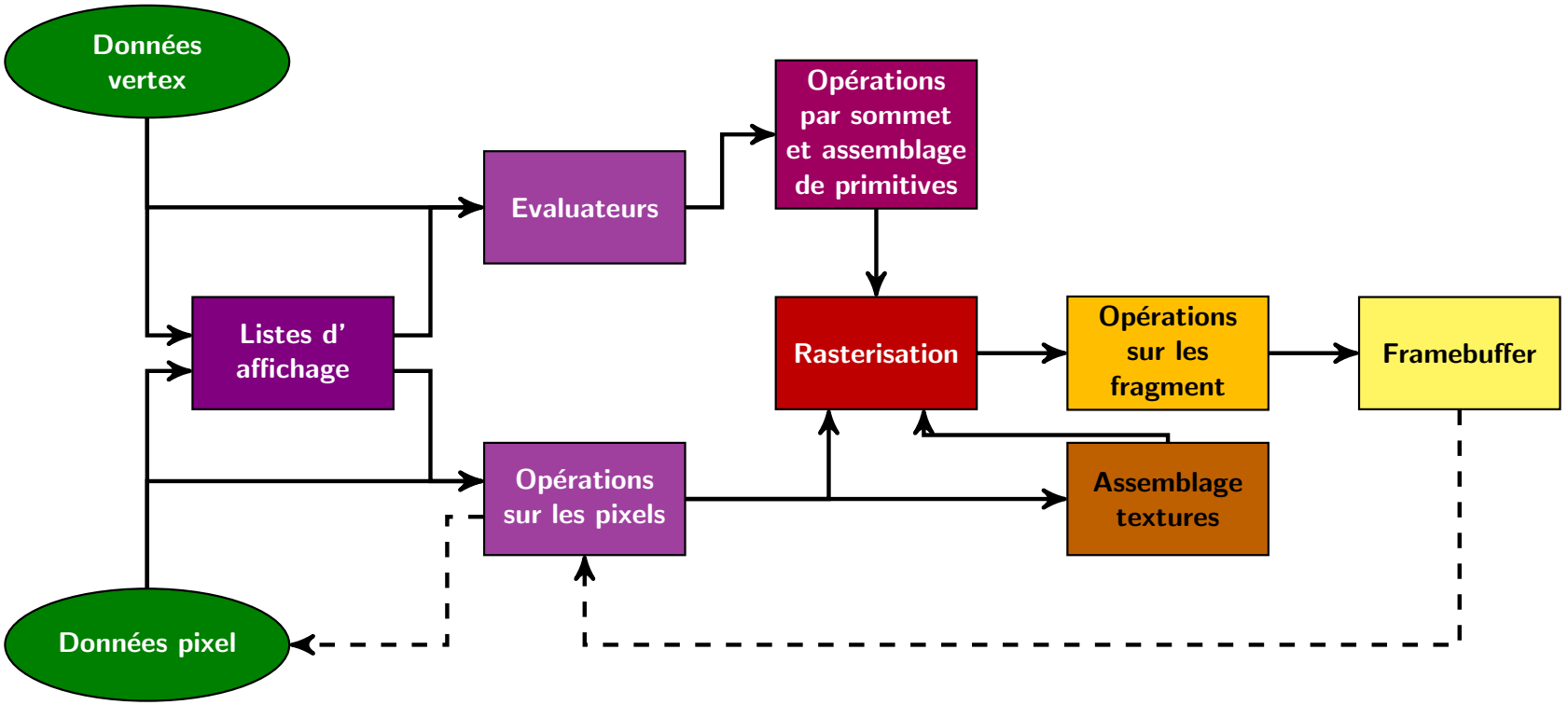
Chronologie comparée



Architecture du pipeline graphique (1/3)

Génération I

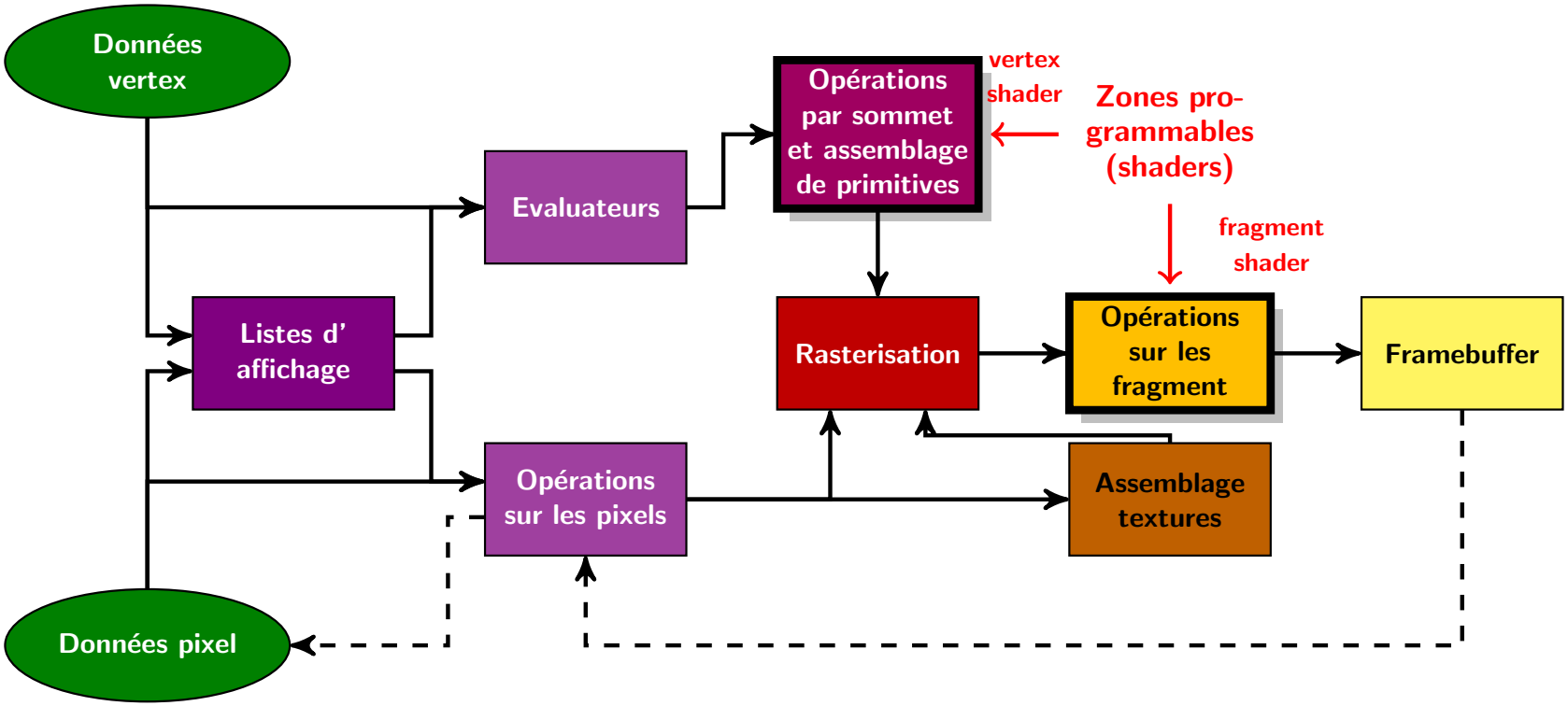
Ensemble de traitements configurables



Architecture du pipeline graphique (2/3)

Génération II

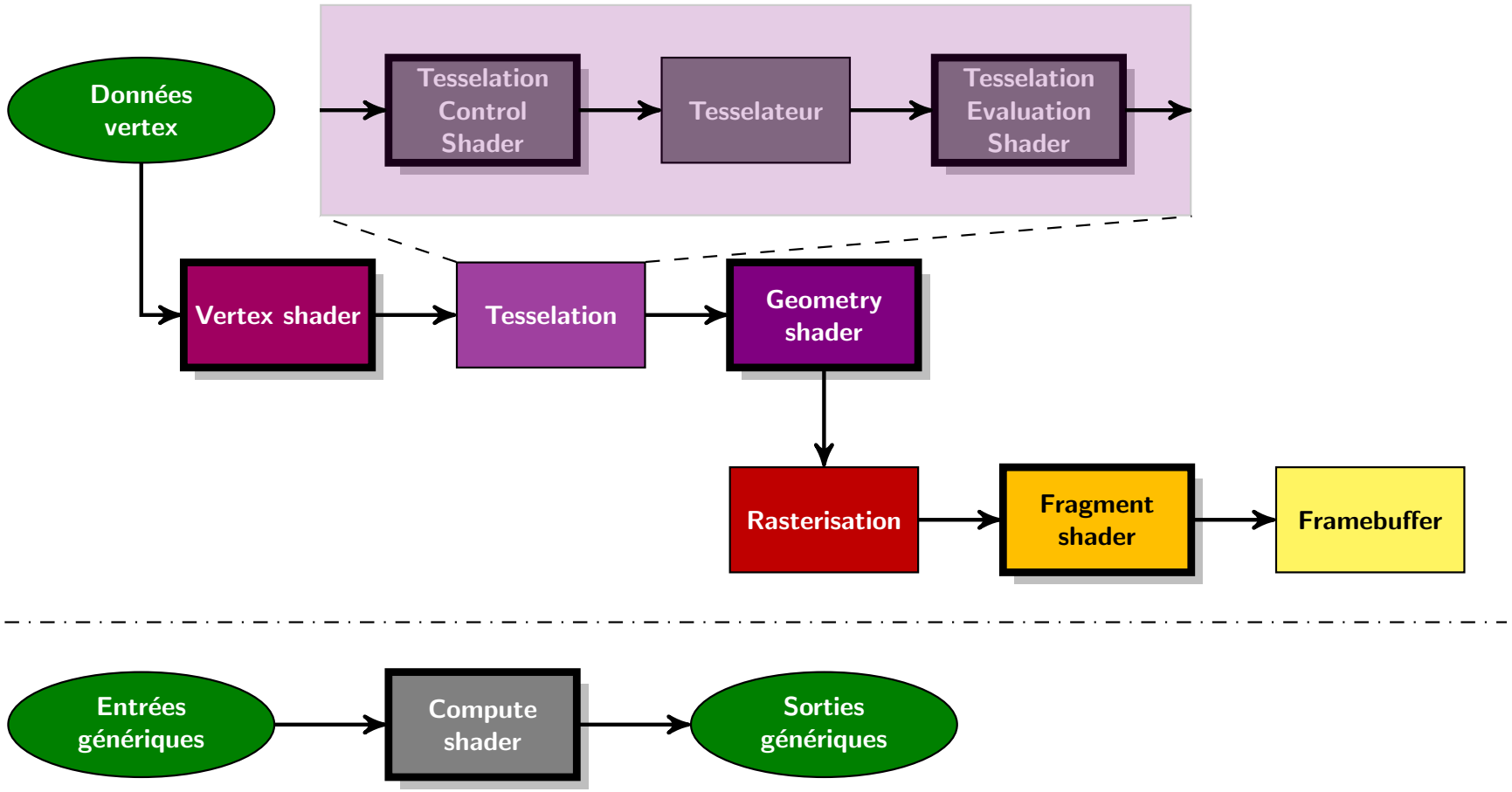
Introduction de traitements programmables (*shaders*) dans le pipeline



Architecture du pipeline graphique (3/3)

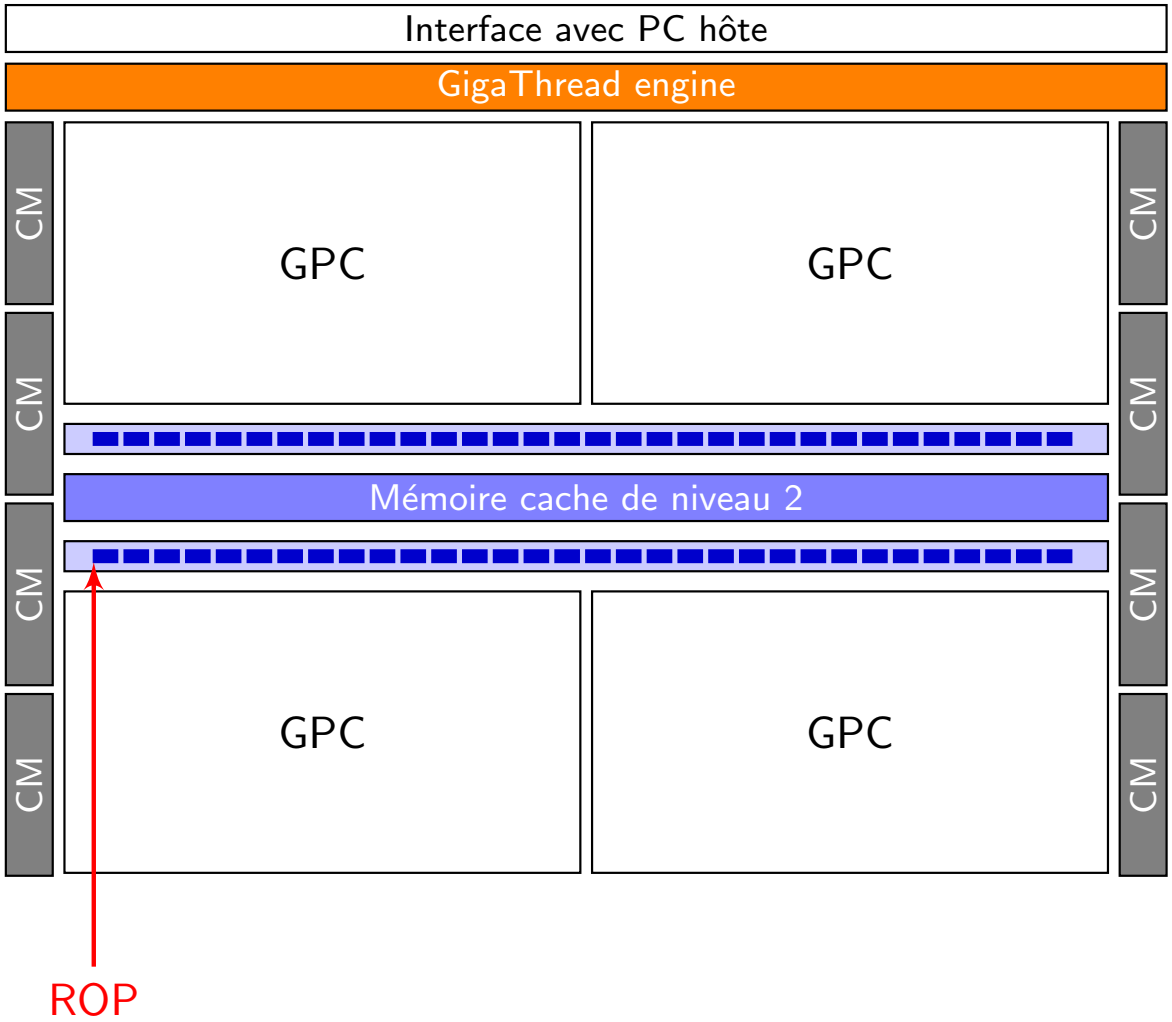
Génération IV

Généralisation des *shaders* et calcul généraliste.



Architecture Pascal (1/4)

NVidia GeForce GTX 1080 – 2016



CM $\times 8$: contrôleur mémoire 32 bits

GPC $\times 4$: *Graphics Processing Cluster*

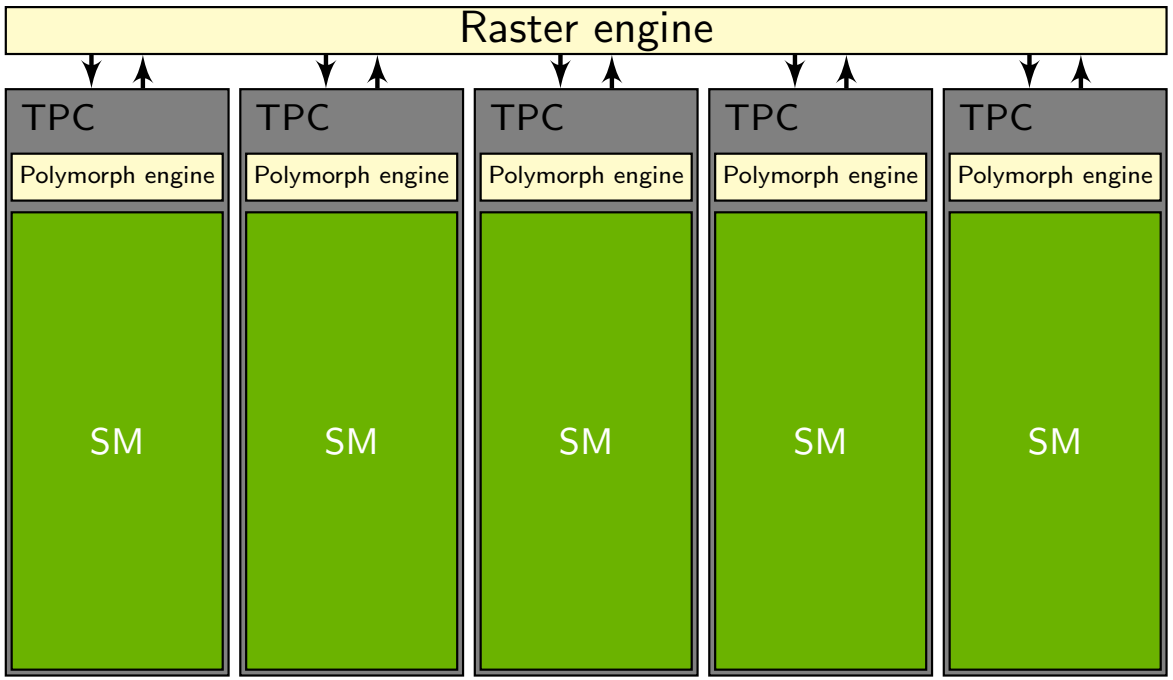
ROP $\times 32$ (*Render Output Unit*) : *blending*, *aliasing*, *z-buffer*, ...

GigaThread engine : affectation des tâches sur les ressources de calcul

Architecture Pascal (2/4)

NVidia GeForce GTX 1080 – 2016

Architecture d'un Graphics Processing Cluster



TPC $\times 5$: *Texture Processing Cluster*

SM $\times 5$: *Streaming Multiprocessor*

Polymorph engine $\times 5$: tesselation, transformations (*shader*), ...

Raster engine : rasterisation

Architecture Pascal (3/4)

NVidia GeForce GTX 1080 – 2016

Architecture d'un *Streaming Multiprocessor*

- Coeurs CUDA 32 bits $\times 128$:
  - ALU (*Arithmetic and logic unit*) ;
  - FPU (*Floating point unit*).
- Load/Store Units $\times 32$: calcul d'adressage (32 threads) ;
- SFU – *Special Function Units* $\times 32$:
  calcul trigonométrique, racine carrée, logarithme . . .
- Unités de texture $\times 8$;
- Registre, 256 kO ;
- Mémoire partagée, 96 kO ;
- Mémoire cache niveau 1, 48 kO.

Architecture Pascal (4/4)

NVidia GeForce GTX 1080 – 2016

Une architecture massivement parrallèle !

- 2560 coeurs CUDA ;
- 160 unités de texture ;
- Remet en cause la vision simpliste du *pipeline* :
  - Motivation pour la génération V !

Récapitulatif rapide

Remarques générales

- Cohabitation de plusieurs générations de cartes graphiques / cohabitation de plusieurs standards et API ;
- Générations de cartes graphiques intriquées avec les API 3D ;
- Architecture massivement parallèle programmables.

Bas niveau / haut niveau

La nécessité d'avoir d'autres API

- OpenGL, Vulkan et Direct3D restent de bas niveau ;
  - Très proches du matériel ;
  - Permettent une optimisation très fine ;
  - A l'usage du **concepteur de moteur de jeu vidéo**.
- Utilisation d'APIs de haut niveau à la place :
  - Gestion de graphe de scène ;
  - Proche de la façon de penser du développeur ;
  - A l'usage du **concepteur de jeu vidéo**.

Architectures parallèles programmables

GPGPU – *General Purpose computing on GPU*

- Utilisation des cartes graphiques pour résoudre des problèmes généralistes : mécanique des fluides, simulation de physique des particules...
- S'appuie sur des compilateurs spécialisés distribuant les calculs entre CPU et GPU.

Exemples d'APIs GPGPU

- OpenCL – *Open Computing Language*, Khronos group ;
- CUDA – *Compute Unified Device Architecture*, NVidia ;
- *Compute shaders*, OpenGL et Direct3D.

Ressources

- Site officiel du consortium Khronos : https://www.khronos.org/
- Guide de programmation de Direct3D 12 : https://msdn.microsoft.com/fr-fr/library/windows/desktop/dn903821
- White paper sur l'architecture Pascal (NVidia) : https://images.nvidia.com/content/pdf/tesla/whitepaper/pascal-architecture-whitepaper.pdf