Introduction à la notion Pipeline Graphique

(initiation)

UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Année universitaire
2004-2005

Pascal Mignot
Pascal.Mignot@univ-reims.fr

UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Objectifs

- **préparation au cours:**
  - d'OpenGL (initiation)
  - de DirectX (programmation multimédia et synthèse d'images)

- **notions introduites:**
  - complément de géométrie
  - pipeline graphique fixe
  - pipeline graphique variable

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Unification de l'écriture des transformations

Transformations courantes en synthèse:

P'=P+V où V est un vecteur (translation)

P'=A.P où A est une matrice 3x3 (rotation, échelle, ...)

P'=d.P/z où P=(x,y,z) et d scalaire (mise en perspective)

problème: comment calculer la composition d'un grand nombre de transformations?

solution: l'espace homogène (4D)

1. Tout point P=(x,y,z) 3D s'écrit dans cet espace P_w=(x,y,z,1).
2. Toutes les transformations T peuvent s'écrire sous la forme:

$$T = \begin{bmatrix} A & B \\ C & 1 \end{bmatrix}$$

A matrice 3x3 (rotation, ...)
B vecteur (translation)
C vecteur ligne (projection)

et représentent la transformation P' = (A.P + B) / (C.P + 1).
En général, C=0 sauf pour la mise en perspective.

3. Composition de transformations = multiplication de matrices 4D.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Écriture et gestion des transformations

Première approche:

- **transformations affines**: P' = A.P + B

$$\begin{bmatrix} P' \\ 1 \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} P \\ 1 \end{bmatrix} = \begin{bmatrix} A.P + B \\ 1 \end{bmatrix}$$

- **composition de transformations**:

  soit P_w un point,

  $P_w' = T_1 . P_w$

  $P_w'' = T_2 . P_w' = T_2 . T_1 . P_w = (T_2 . T_1) . P_w$

  La transformation finale est $T = T_2 . T_1$ (ordre de droite à gauche).

  On peut ainsi accumuler dans une seule matrice 4D un nombre quelconque de transformations, et l'utiliser pour transformer des points 3D.

Les cartes graphiques gèrent toutes les transformations dans l'espace homogènes.

Détails:

et bientôt vous aussi ... au second semestre.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Couleur et alpha

Les couleurs sont codées sous forme d'un triplet RGB=(rouge,vert,bleu).

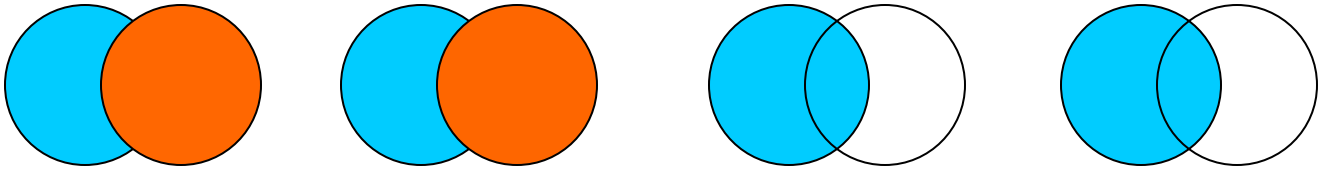
On introduit en plus la notion de transparence (appelé alpha α) .

α=0 : couleur transparente.

α=1 : couleur opaque.

Le mélange d'une couleur C_1 est mélangée avec une couleur C_2 de transparence α est calculée avec:

$$C = \alpha.C_2 + (1-\alpha).C_1$$



α=1 **α=0.66** **α=0.33** **α=0**

Le alpha peut être utilisé pour d'autres types de mélange.

Les couleurs sur une carte 3D peuvent être codées sur 16 bits (R5G6B5, A1R5G5B5), 24 bits (R8G8B8), 32 bits (A8R8G8B8, A2R10G10B10).

Pipeline Graphique
Définition

De façon généraliste:

suite des traitements appliqués à un objet (ou à un ensemble d'objets) géométrique(s) dans le but d'en obtenir le rendu.

En entrée:

un objet géométrique 3D (propriétés géométriques et matérielles).

En sortie:

une image (2D).

Sur une carte graphique:

objet 3D = défini par des facettes

rendu 3D = avec un z-buffer

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Pipeline Graphique
Entrée du pipeline

- définition d'un tableau de facettes:
  - tableau de sommets:
    groupe de 3 sommets = une facette
    l'ordre des sommets oriente la facette
  - tableau de sommets indexés:
    tableau de sommets + tableau d'index
    l'ordre des indices oriente la facette

- définition d'un sommet:
  sommet = structure (au sens struct du C).
  - géométrie: position, normale,
  - couleur diffuse, spéculaire.
  - coordonnées de « texture » : pour accès à des fonctionnelles 1D à 4D.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Pipeline graphique

les 3 étapes

- **Pipeline Géométrique**

  traitement de chaque sommet.

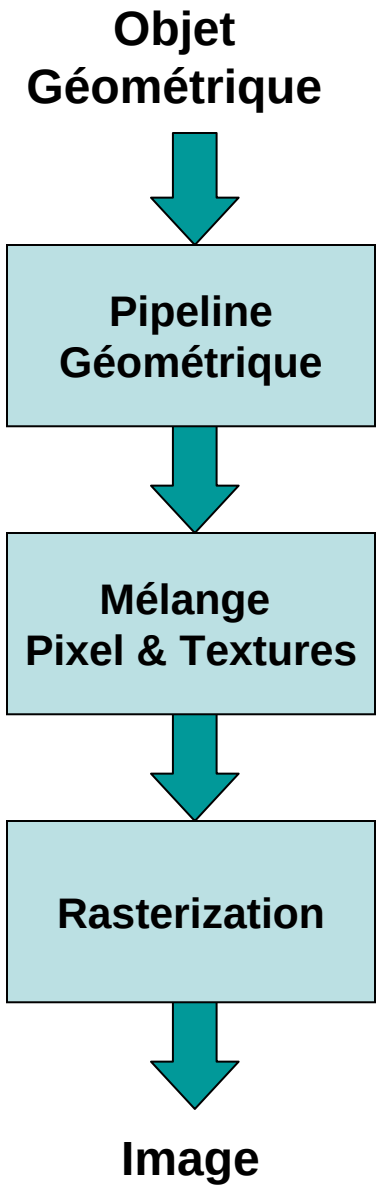
- **Mélange Pixel-Textures**

  calcul de la valeur des pixels associée à la facette.

- **Rasterization**

  mélange entre les pixels de la facette et ceux déjà présents sur l'écran.

Plus de détail sur ces étapes.



UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Pipeline graphique
sortie du pipeline

Note: l'utilisation d'un seul tampon génère des déchirements de l'image (Tearing) si le tampon est affiché à l'écran au moment où on est entrain de le mettre à jour (csq: mélange des deux images).

⇨ sinon, on est obligé de générer les images au même rythme que le vitesse de rafraîchissement de l'écran.

L'affichage correct d'une image nécessite donc au moins deux tampons:

- le premier contient ce qui est actuellement affiché à l'écran (FrontBuffer).
- le second destiné à accueillir l'image suivante à afficher (BackBuffer).

Le rafraîchissement de l'image affichée se fait: soit par échange (swap: le Frontbuffer devient le Backbuffer et vice-versa), soit par copie (réalisable entre deux rafraichissements).

UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Pipeline graphique

Pipeline Géométrique

Pour chaque sommet:

- transformation géométrique des sommets:
  - changement de repère (observateur, repère local)
  - mise en perspective et projection
- évaluation de quantités:
  - luminance,
  - génération de coordonnées de texture (texgen),
  - ...

Ce calcul a lieu en chaque sommet et indépendamment les uns des autres.

A la fin de cette phase,

- les sommets sont assemblés en facettes (=triangle setup).
- pour chaque facette, on génère la liste des pixels associés sur l'écran.

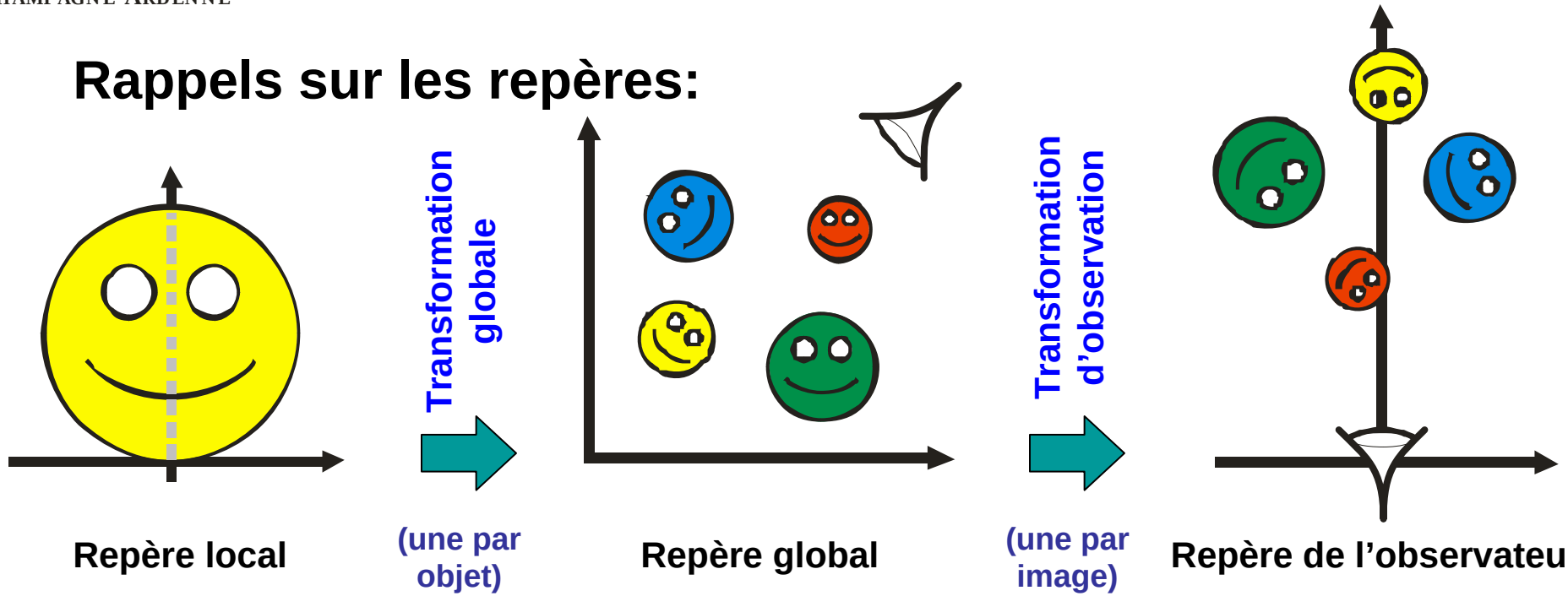
Remarque: cette étape du pipeline graphique est aussi connue sous le nom de TnL (Transformation aNd Lighting).

UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Pipeline graphique

Pipeline Géométrique

Rappels sur les repères:



UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Pipeline graphique

Pipeline Géométrique

Transformations géométriques:

- **transformation globale:**
  cette transformation amène l'objet depuis son repère local dans le repère global de la scène.
- **transformation d'observation:**
  cette transformation passe du repère global au repère de l'observateur (aligne l'axe de vision avec l'axe z).
- **transformation de projection:**
  cette transformation effectue la mise en perspective et la projection dans le plan de l'écran.

La composition de ces trois transformation amène un objet de son repère local dans le repère de l'écran.

Evaluation de la luminance:

Le calcul de l'éclairage se fait (par défaut) avec le modèle de rendu de Blinn-Phong (=modèle de rendu de Phong légèrement modifié).

On calcule la luminance diffuse et spéculaire (indépendamment).

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Pipeline graphique

Mélange Pixel & Textures

Pour chaque pixel de chaque facette,

- interpolation au pixel des valeurs aux sommets de la facette (couleurs, coordonnées de texture, ...).

  par défaut, ombrage de Gouraud.

- avec l'échantillonneur, lecture des textures aux coordonnées interpolées.
- mélange des couleurs diffuses, spéculaires et de la (ou des) texture(s).

  addition (ajout de contribution), multiplication (pondération de contribution), ...

Une fois ce calcul fait, on obtient un pixel texturé et éclairé (= un texel).

Pour l'ensemble des pixels d'une facette, on obtient à la fin de cette étape, l'ensemble des texels tels qu'ils seraient affichés indépendamment de ce qui est déjà présent à l'écran.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Pipeline graphique

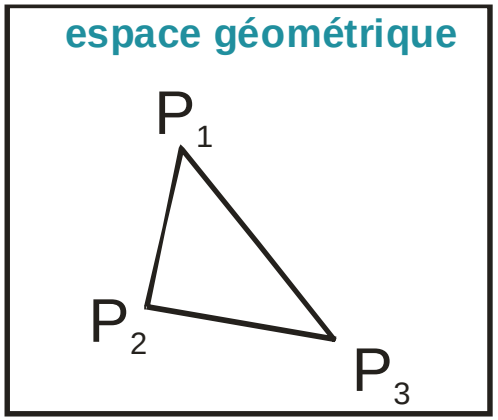
Mélange Pixel & Textures

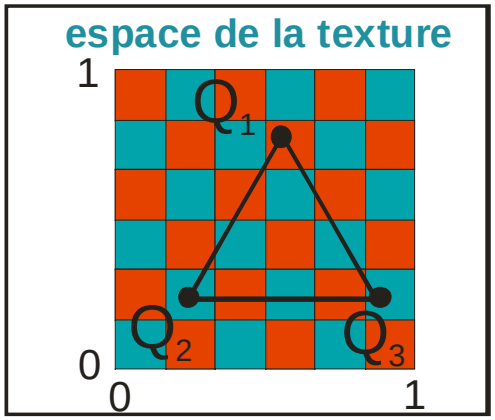
Principe du texturage

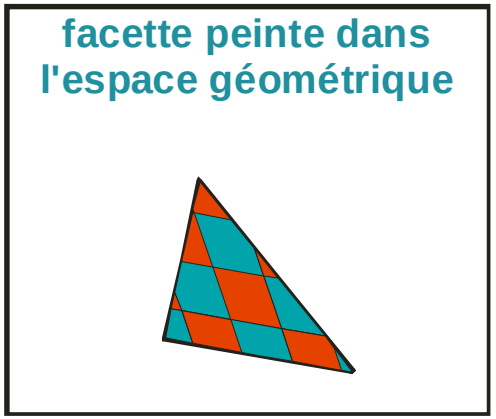
- Des coordonnées de texture Qi associées à chaque sommet Pi sont absolument nécessaires.

  note: il est possible d'avoir dans la structure de sommets plusieurs coordonnées de texture différentes.
- En chaque pixel, les coordonnées de texture sont interpolées à partir des valeurs aux sommets.
- La texture peinte sur la facette (P_1, P_2, P_3) et celle contenue dans le triangle (Q_1, Q_2, Q_3) de l'espace de la texture.

Déformation de la texture:
- si les coordonnées sont mal choisies.
- si la texture est inadaptée à la surface.



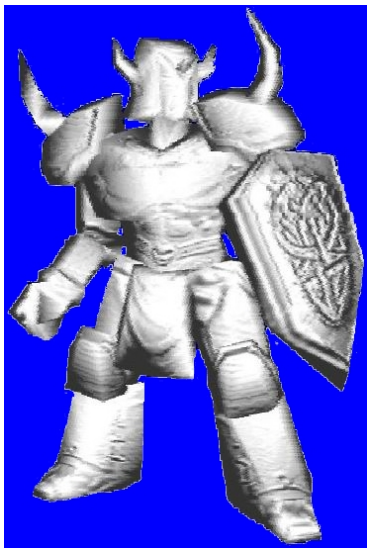




UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Pipeline graphique

Mélange Pixel & Textures : exemple




(diffus x décalage)+(spéculaire x brillance)=

UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Pipeline graphique
Rasterization

Phase durant laquelle les texels de chaque facette sont mélangés avec les pixels déjà présent sur l'image de rendu (backbuffer).

Cette phase fait intervenir:

- le alpha
- le tampon de profondeur (z-buffer)
- le pochoir (stencil buffer), *voir ci-après*

A partir de ceux-ci ont détermine:

- s'il faut mettre à jour le backbuffer.
  en fonction du z-buffer et du stencil buffer au pixel courant.
- comment le mettre à jour.
  en fonction de la valeur de alpha du texel et de l'opération de mise à jour.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Pipeline graphique
Rasterization

Qu'est-ce qu'un pochoir?

- buffer supplémentaire (4 ou 8 bits) directement associé au z-buffer.
- son contenu peut être utilisé pour modifier le comportement du z-buffer.
- un test spécifique est attaché au pochoir.

Utilisation du z-stencil buffer

- **opérations**

```
si stencil_test
alors si z_test
          alors  stencil_success_operation
          sinon  stencil_zfail_operation
sinon            stencil_fail_operation
```

- les 3 opérations différentes de mise-à-jour du stencil (stencil_*_operation) peuvent être effectuée en fonction des cas.

Exemple d'utilisation du stencil-buffer

- compteur de complexité d'un pixel
- masque d'affichage
- gestion des ombres par la méthode des shadow volumes (double sided stencil) ...

UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Pipeline graphique
Rasterization

Lien entre le z-buffer et le stencil buffer

le test du z-buffer n'est effectué que si le test du stencil à réussi.

Mise à jour du backbuffer

n'a lieu que si le test du z-buffer à réussi

dans ce cas, le backbuffer et le zbuffer sont mis à jour*.

à partir de:

- la valeur du texel calculée (alpha compris),
- la valeur déjà contenue dans le backbuffer,
- l'opération de mélange définie.

pour calculer la valeur qui sera écrite dans le backbuffer.

Exemples: texel = (α, C_2) , backbuffer= C_1

C = nouvelle valeur écrite dans le backbuffer

$C = \alpha.C_2 + (1-\alpha).C_1$

$C = C_2$

* il est possible de désactiver (temporairement) ces mises-à-jour.

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Pipeline Graphique Avancé

Pipeline graphique programmable

Depuis quelques années, les cartes graphiques deviennent programmables:

- **pipeline géométrique** (transformation & TnL)
  remplacés par un code écrit par l'utilisateur.
  $\Rightarrow$ vertex shader
- **mélange pixel & textures**
  remplacés par un code écrit par l'utilisateur.
  $\Rightarrow$ pixel shader

Applications:

- permet d'améliorer le rendu:
  ombrages avancés: ombrage de Phong, cartoon shading, ...
  transformations complexes ou dynamiques
- multiplication des applications sur une carte vidéo 3D:
  traitement d'images sur des flux vidéo,
  calculs numériques, ...

Modèles de shader
et cartes graphiques

Modèles de shader:

SM 1.* :

capacité de programmation très limités.
pas de contrôle de flux (pas de tests/boucles).
textures difficiles d'utilisation.

SM 2.* :

capacités de programmation étendues (256-512 instructions).
programmation étendue des textures.
contrôle de flux statique.

SM 3.* :

capacités de programmation plus étendue (>512 instructions).
textures utilisables dans un VS.
contrôle de flux dynamique (tests et boucles).

Cartes graphiques associées:

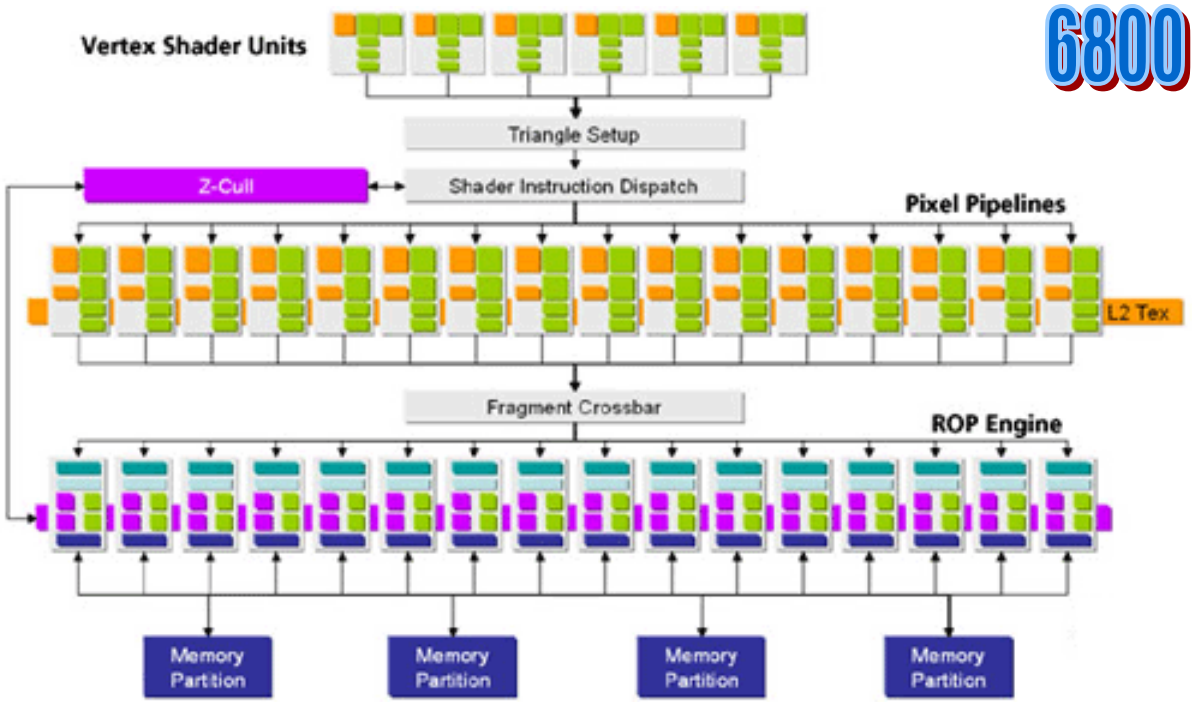
2.0 (radeon 9800, X600, X800)
2.0x (nvidia FX 5xxx)
3.0 (nVidia 6800, 6600).

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Pipeline Graphique Avancé

Pipeline graphique programmable

Exemple du pipeline graphique de la GeForce 6800



Vertex Shader Unit : traitement et transformation des sommets.

Triangle Setup : construction et répartition des pixels associés à chaque facette.

Shader Instruction Dispatch : répartition des instructions du PS entre les différents pixel pipelines.

Pixel Shader Unit : traitement et texturage des pixels.

Fragment crossbar : assignation de la sortie des PS à un ROP disponible.

ROP units : opérations de rasterization et de sortie.

Z-Cull : Z/stencil

Memory Partition : RenderTargets

UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Pipeline Graphique Avancé
Tesselator

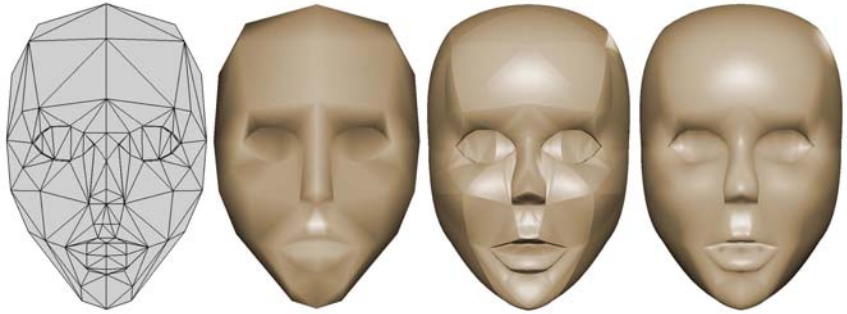
Les cartes graphiques récentes commencent à supporter les primitives de haut niveaux (de type bézier):

- RT-patches (Nvidia)
- N-patches (ATI, Matrox)

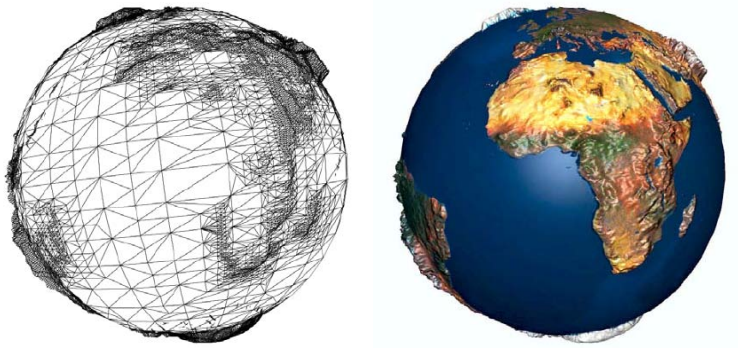
Le tesselator créé automatiquement les facettes associées à la primitive (entrée + triangle setup).

Autres applications:

- **subdivision:** augmentation automatique de la précision d'une surface en ajoutant des sommets.



- **displacement mapping:** utilisation d'une texture pour déformer une surface.



UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Programmation de cartes graphiques

- **à un niveau élémentaire,**

  permet de représenter simplement des objets ou des environnements 3D.

  difficulté = contrôle de l'interface (DirectX ou OpenGL).

  cours d'introduction à OpenGL (le tout début).

  cours de programmation multimédia (3D + son + vidéo).

- **à un niveau avancé,**

  permet d'effectuer des images 3D « qualité cinéma ».

  difficulté = concept de synthèse d'images pour l'implémentation des shaders correspondants.

  cours de synthèse d'images.

UNIVERSITÉ
DE REIMS
CHAMPAGNE-ARDENNE

Fin?

- **Fin des cours théoriques de l'initiation à la synthèse**
- **Début du cours d'OpenGL:**
  la semaine prochaine.
- **Projet:**
  - implémentation d'un z-buffer avec rendu de Gouraud et Phong (voir le site).
  - projets OpenGL (cf l'intervenant).

  soutenances en janvier.
- **Examen:**
  - Une séance de préparation à l'examen la dernière semaine de décembre.
  - Le cours est autorisés à l'examen (polycopié de cours seulement, pas les corrections de TDs).