

Synthèse d'images et modélisation géométrique

Introduction à la synthèse et à OPENGL

Nicholas Journet

26 février 2013

Plan

Introduction

Chaîne de
synthèse

Introduction
OpenGL

- ▶ Introduction
- ▶ Chaîne de synthèse
- ▶ Introduction à OpenGL

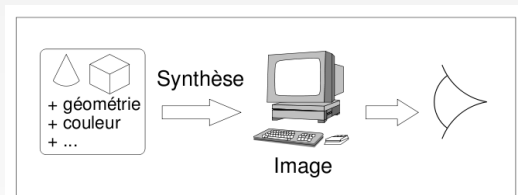
Bibliographie

- ▶ Cours de Synthèse d'images et modélisation géométrique - G Thomas - Université de Bordeaux
- ▶ Introduction à OpenGL - X Michelon - Linuxorg Cours en ligne

Synthèse

Synthèse d'image :

- ▶ Entrée : description géométrique d'une scène
- ▶ Sortie : une image de la scène la plus réaliste possible



Difficultés

- ▶ Modéliser la scène avec des primitives géométriques
- ▶ Comprendre les mécanismes de la vision, pour rendre l'image réaliste

Applications

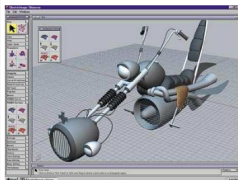
Introduction

Chaîne de synthèse

Introduction OpenGL

Cinéma, Marketing, jeux, Réalité virtuelle

- ▶ Domaine : Modélisation géométrique, rendu, animation...
- ▶ Contraintes : Réalisme et rapidité
- ▶ Programmes : 3DS max, Maya...



Applications

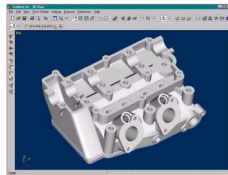
Introduction

Chaîne de synthèse

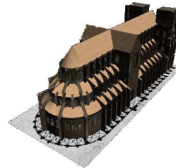
Introduction OpenGL

Conception automobile, navale, architecturale

- ▶ Domaine : Modélisation géométrique
- ▶ Contraintes : Expressivité, détails et rapidité
- ▶ Programmes : Catia, autoCAD...



Logiciel Catia



Modélisation de Notre Dame de
Paris

Applications

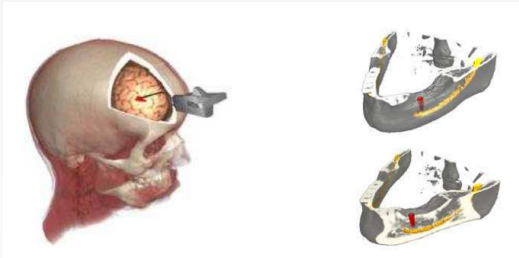
Introduction

Chaîne de synthèse

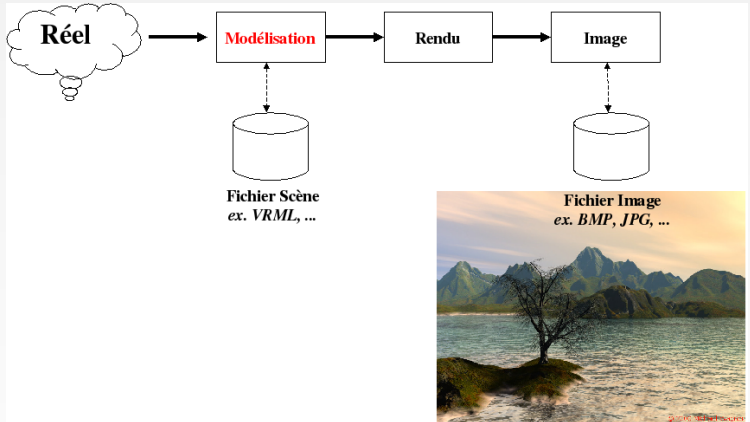
Introduction OpenGL

Biomédical (imagerie + chirurgie assistée)

- ▶ Domaine : analyse, visualisation, mesure, modélisation, géométrie
- ▶ Contraintes : fiabilité, réalisme, rapidité.



Chaîne de synthèse



Modèle géométrique

- ▶ Primitives : sphères, cylindres, cubes
- ▶ Constructive Solid Geometry
- ▶ Surfaces à base de facettes polygonales (ou Maillages)
- ▶ Représentations paramétriques (Bézier, B-Splines, Nurbs)

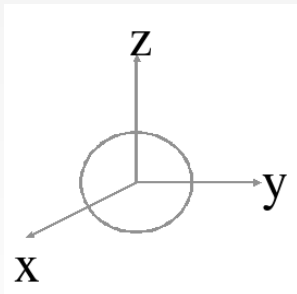
Primitives

Introduction

Chaîne de
synthèse

Introduction
OpenGL

- ▶ Sphere (X_o, Y_o, Z_o, R)
- ▶ Boite (X_o, Y_o, Z_o, W, L, H)
- ▶ Cône (X_o, Y_o, Z_o, R, r, H)



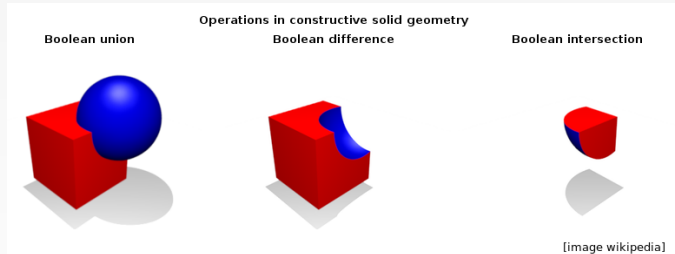
Constructive Solid Geometry

Introduction

Chaîne de synthèse

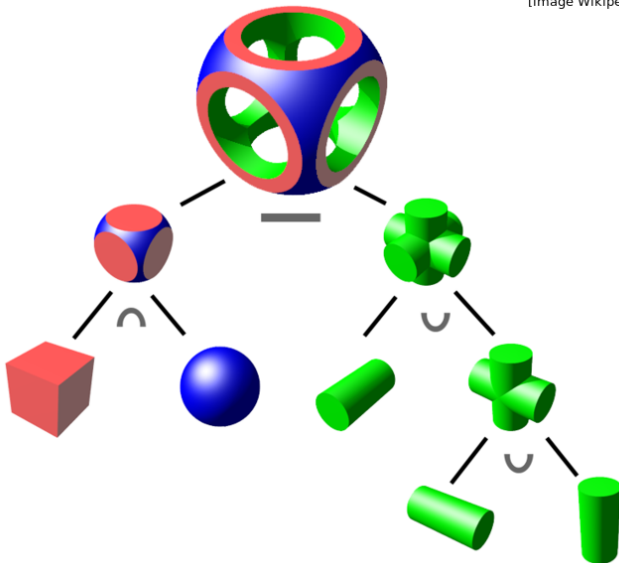
Introduction OpenGL

- ▶ Générer des formes complexes à l'aide de primitives.
- ▶ Dessiner un objet : rogner des parties, percer des trous,...
- ▶ Coller des pièces entre-elles
- ▶ Utilisé généralement dans la CAO.



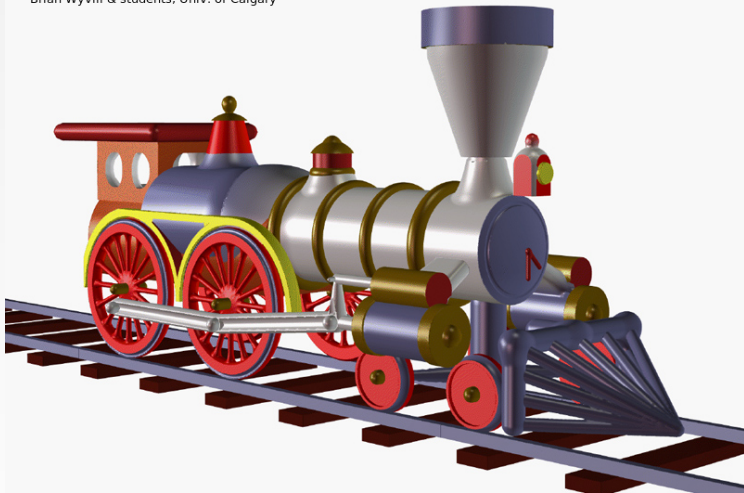
Constructive Solid Geometry

[image Wikipedia]



Un train en CSG

Brian Wyvill & students, Univ. of Calgary



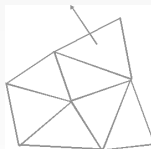
Introduction

Chaîne de
synthèse

Introduction
OpenGL

Maillage

- ▶ Modélisation polygonale : le modèle est assimilé à un ensemble de polygones (liste de sommets et d'arêtes).
- ▶ La normale donne l'orientation de la facette (différencier l'extérieur et l'intérieur)
- ▶ Sans effet de lissage, l'objet apparaîtra anguleux si la définition en facettes est faible.
- ▶ C'est la technique majoritairement utilisée dans le jeu vidéo

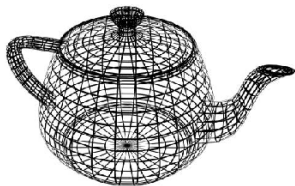


Maillage

Introduction

Chaîne de
synthèse

Introduction
OpenGL

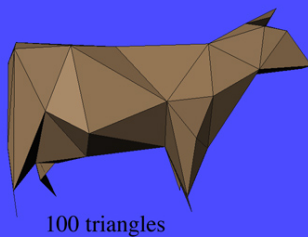
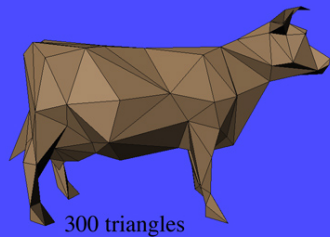
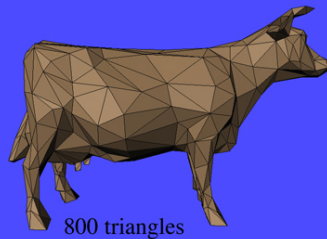
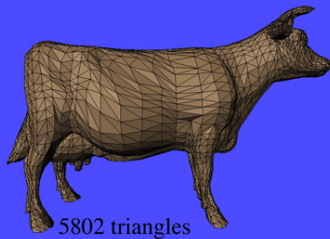


Maillage

Introduction

Chaîne de
synthèse

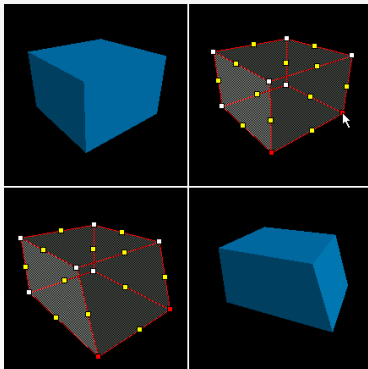
Introduction
OpenGL



Modélisation par courbes

Principe :

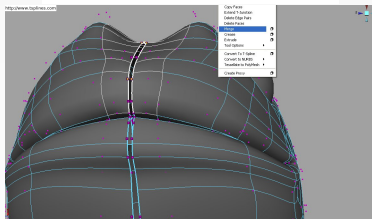
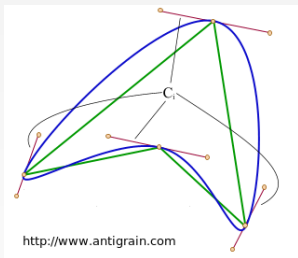
- ▶ dessiner quelques choses de courbes (lisses et continues)
- ▶ édition locale : retouches ponctuelles, influence limitée



Modélisation par courbes

Solution Générale

- ▶ courbes paramétriques polynomiales (par morceaux)
- ▶ définies par des points de contrôle (enveloppe convexe)
- ▶ Modèles : Bézier, B-Spline, Nurbs



OpenGL : Open Graphics Library

- ▶ API graphique
 - ▶ Couche entre le programmeur et le matériel (ou d'autres programmes)
- ▶ Environ 250 procédures et fonctions
 - ▶ Définition des objets
 - ▶ Opérations pour applications interactives

Présentation OpenGL

- ▶ Développé par SGI au début des années 90
- ▶ SGI n'est plus propriétaire : license gratuite
- ▶ Evolution contrôlée (architecture review board)
Microsoft (plus depuis 2003), Dell, IBM, Intel, Matrox, ATI,...
- ▶ Largement utilisé et maintenu
- ▶ Très bien documenté : www.opengl.org
- ▶ Facile à utiliser

OpenGL est utilisé pour

- ▶ Applications temps réel (3D Studio Max, Maya, ...)
- ▶ Environnements virtuels interactifs (ubuntu)
- ▶ Jeux videos (Quake, Warcraft 3, Medal of Honor, ...)



Fonctionnement de OpenGL

Interprétation client / serveur

1. Le programme (client) invoque des commandes (Eg. activation des lumières, rendu de triangles, ...)
2. Les commandes sont interprétées et traitées par le serveur "GL"
 - ▶ OpenGL ne fournit pas le moyen de construire des scènes complexes (utiliser des API plus haut-niveaux : Java3D, OpenInventor,...)
 - ▶ Ne gère pas l'IHM (il faut utiliser la GLUT)

Primitive géométrique d'OpenGL



Points



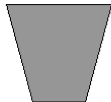
Ligne



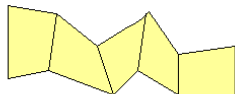
Polygone



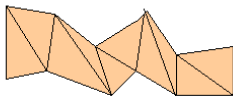
Triangle



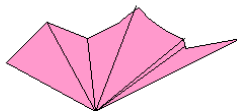
Quad



Quad strip



Triangle strip



Triangle Fan

Rendu d'une primitive géométrique

- ▶ Dans un tampon d'images
- ▶ Primitives OpenGL
 - ▶ Un ensemble de sommets
 - ▶ Un sommet définit : un point, une extrémité d'un segment, le sommet d'un polygone

Rendu OpenGL :

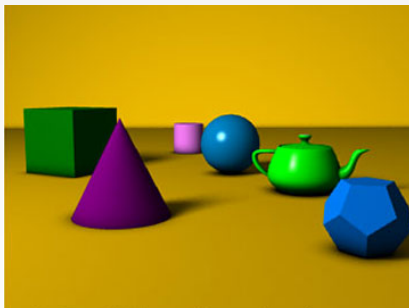
- ▶ Données associées à un sommet
 1. coordonnées
 2. couleur
 3. normale
 4. coordonnées de texture

Modes de rendu

1. Fil de fer (Wireframe)
2. Plat (Flat Shading) : une seule couleur par polygone
3. Interpolé : les couleurs des sommets des polygones sont interpolées
4. plaquage de texture

Elimination des parties cachées

- ▶ A chaque fois qu'un fragment i est dessiné, le z_i (distance au point de vue) est comparé et peut-être stocké dans le tampon de profondeur (Z-buffer)
- ▶ Soit z_j la valeur présente dans le z-buffer
 - ▶ Si $z_j > z_i$ le fragment est dessiné
 - ▶ Sinon rien n'est fait



Modèles de couleurs avec OpenGL

- ▶ RGBA
 - ▶ Red, Green, Blue, Alpha
 - ▶ Un canal pour chaque couleur
 - ▶ 8 bits/canal = 16 million de couleurs
- ▶ Couleur indexée (Indexed Color) : un petit nombre de couleurs accédées grâce à un indice dans une table de couleurs

Transparence avec OpenGL

- ▶ Utilisation d'un modèle RGBA, ma 4ème composante (alpha) spécifie la transparence
 - ▶ $\alpha = 0$; polygone complètement transparent
 - ▶ $\alpha = 1$; polygone opaque
- ▶ Deux objets de couleurs (C_s, C_f) sont composés au moment du rendu
 - ▶ $C = \alpha * C_s + (1 - \alpha)C_f$
 - ▶ C_s est la couleur du nouveau fragment transparent
 - ▶ C_f est la couleur déjà présente dans la mémoire tampon

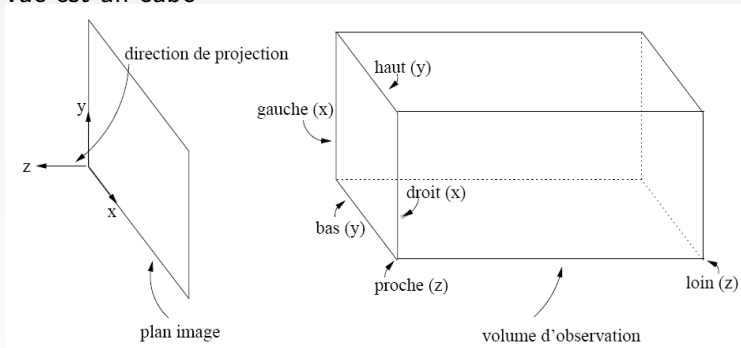
Visualisation

Introduction

Chaîne de synthèse

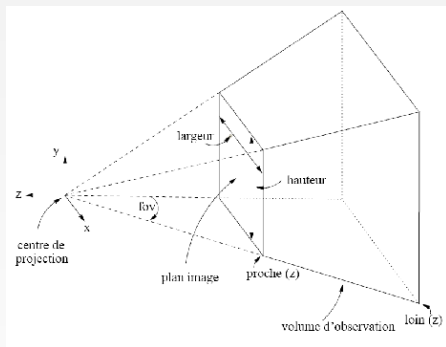
Introduction OpenGL

Projection orthographique : Projection parallèle, le volume de vue est un cube



Visualisation

Projection perspective : Le volume de vue et de découpage est une pyramide



Position caméra

La caméra par défaut est toujours située à l'origine et pointe vers la direction des z négatifs