

Durée : 5 jour(s)

Objectifs

Appréhender l'API et les concepts importants d'OpenGL, ainsi que les particularités de la 3D temps réel, autant avec le pipeline fixe qu'avec les shaders

Découvrir l'étendue des fonctionnalités d'OpenGL.

Pré-requis

Compétences en langage C, quelques notions concernant le monde de la 3D.

Plan de cours

Présentation

versions et historique (1.x à 4.x, ES1, ES2)
place d'OpenGL sur le marché actuel de la 3D
principes de fonctionnement d'une carte 3D
pipeline fixe et pipeline programmable
extensions OpenGL
bindings et langages

Initialisation et contexte

création de l'espace de rendu
les API concernées : GLX, WGL, CGL, EGL, ...
les abstractions possibles : GLUT, SDL, ...
gestion des extensions (GLEW, GLEE, ...)
le cas de l'API GLU

Principes de base

définition d'une scène dans un espace en 3D
états de la machine OpenGL
espace de visualisation : Frustum

Formes, volumes et géométries

points, lignes et polygones
concepts : les surfaces évaluées (Bézier) et les NURBS de GLU
géométries arbitraires
performances et triangles
mode immédiat, listes d'affichages, Vertex Array, VertexBuffers

Matrices

Rôle des matrices de la machine OpenGL
Matrice de visualisation
Matrice de transformation
Rotations, translations

Eclairage

Rôle et fonctionnement de l'éclairage
Simplifications du modèle d'éclairage
Mise en place et définitions

Déplacements des sources lumineuses

Gestion des couleurs

Gestion des matériaux

Les normales (déduction et lissage)

Les spots

Le blending et les transparences

intérêts et problématique du blending

problématiques des superpositions blendées

Application de textures

Principes du texturage

Chargement de textures

Mise en place de coordonnées de texture

Filtrages (linéaires, bilinéaires)

MipMapping

Matrice de texturage

Extensions (multitexturing, textures 3D, ...)

Précisions sur le blending de textures

Tampons

Tampon de profondeur (Z-buffer)

Tampon d'accumulation

Tampon "pochoir" (stencil buffer)

Framebuffer Objects (FBO)

Utilisations avancées des tampons (réflexions, blur, stencil shadows, cell shading, ...)

Shaders

présentation

Vertex Shaders et Fragment Shaders

Geometry Shaders (OpenGL 3.2) et tessellation (4.0)

compilation et édition des liens des shaders

le langage GLSL

types, passages d'arguments, ...

branchements et itérations

mise en oeuvre (toon shaders, normal mapping, post-processing, ...)

Durée :

Objectifs

Pré-requis

Plan de cours

Réalisme d'une scène
ombres
gestion du brouillard
antialiasing
skyboxes, dômes, ...
particules et impostors
gestion temporelle
textures animées
physique d'un environnement 3D
textures animées, render-to-texture (RTT)
gestion des entrées utilisateur
workflow de création et gestion des assets
performance et mémoire

Présentation du GPGPU
concepts de calcul embarqué dans le GPU
intérêts et contraintes
Shaders et FBO
OpenCL (ouvert)
CUDA (Nvidia)