



OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

# OpenGL

## Géométrie et Prise de Vue

Didier Verna

[didier@lrde.epita.fr](mailto:didier@lrde.epita.fr)  
<http://www.lrde.epita.fr/~didier>

Version ENSTA – D9-1 du 16 novembre 2009



# License d'exploitation

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

Ce document est mis à votre disposition sous un contrat de license Creative Commons – Common Deed. Par le téléchargement ou la consultation de ce document, l'utilisateur accepte les conditions d'utilisation décrites par cette license et s'engage à la respecter intégralement.

La licence confère à l'utilisateur un droit d'usage sur le document consulté ou téléchargé, totalement ou en partie, dans les conditions définies ci-après et à l'exclusion expresse de toute utilisation commerciale.

Le droit d'usage défini par la licence autorise un usage à destination de tout public qui comprend:

- le droit de reproduire tout ou partie du document sur support informatique ou papier,
- le droit de diffuser tout ou partie du document au public sur support papier ou informatique, y compris mise à la disposition du public sur un réseau numérique.

Aucune modification du document dans son contenu, sa forme ou sa présentation n'est autorisée. Les mentions relatives à la source du document et/ou à son auteur doivent être conservées dans leur intégralité.

Le droit d'usage défini par la licence est personnel, non exclusif et non transmissible. Tout autre usage que ceux prévus par la licence est soumis à autorisation préalable et expresse de l'auteur.





# Table des matières

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

- 1 Composition d'objets**
  - Primitives Géométriques
  - Suppression de surfaces
- 2 Composition d'image**
  - Placement et transformation des objets
  - Placement du point de vue
  - Volume de projection
  - Viewport
- 3 Résultat**



# Modélisation géométrique

La « facette » cachée des objets

OpenGL

Didier Verna  
ENSTA  
D9-1

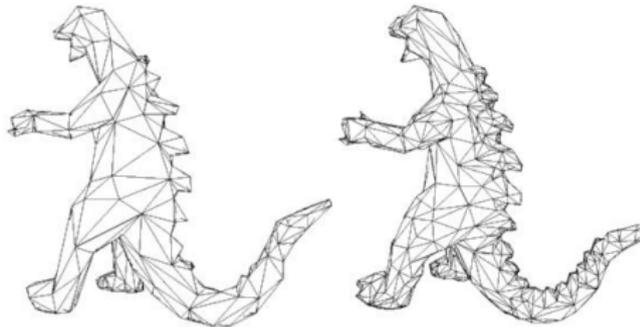
Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat



- **Modélisation** : décomposition en « primitives »
- **Primitives** : point, lignes, polygones (triangles)
- **Spécification** : par sommets (vertex, *plur.* vertices)



# Composition d'objets en OpenGL

À coup de primitives

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

- Primitives OpenGL : points, lignes, polygones
- Ligne OpenGL : segment de droite
- Polygone OpenGL : polygone simple et convexe
- Vertex :  $(x, y, z, w) \implies (x/w, y/w, z/w)$  dans l'espace euclidien

**Pour les gens pressés...**

```
void glRect{sfid} (TYPE x1, TYPE y1, TYPE x2, TYPE y2);
```

```
void glRect{sfid}v (TYPE *v1, TYPE *v2);
```

Tracent un rectangle défini par une diagonale dans le plan  $(X, Y)$ .



# Mécanisme général

Composition de formes par liste de vertices

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

**void** glBegin (GLenum mode);

Démarre la composition de primitive(s) géométrique(s).

*mode* : type de primitive(s)

**void** glVertex{234}{sifd}[v] (TYPE coords);

Définit un nouveau vertex composant la primitive courante.

**void** glEnd (**void**);

Termine la composition.

```
glBegin (GL_POLYGON);  
{  
    glVertex3f (5.0, -2.0, 0.0);  
    // ...  
}  
glEnd ();
```



# Modes de composition

Ils y sont tous

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives

Surfaces cachées

Scène

Objets

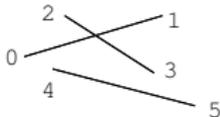
Point de vue

Projection

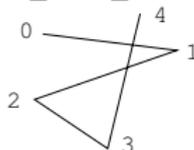
Viewport

Résultat

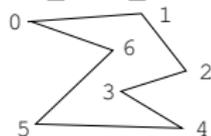
**GL\_LINES**



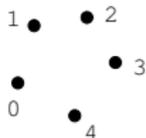
**GL\_LINE\_STRIP**



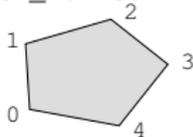
**GL\_LINE\_LOOP**



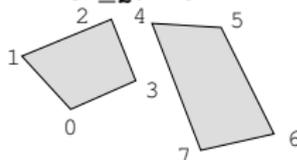
**GL\_POINTS**



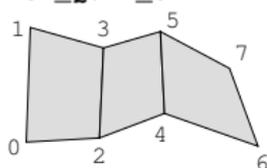
**GL\_POLYGON**



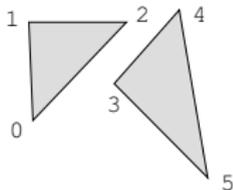
**GL\_QUADS**



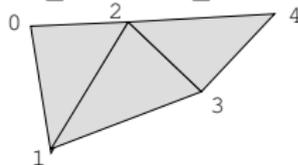
**GL\_QUAD\_STRIP**



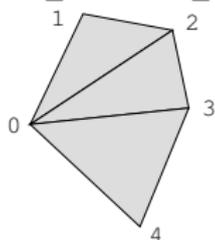
**GL\_TRIANGLES**



**GL\_TRIANGLE\_STRIP**



**GL\_TRIANGLE\_FAN**





# Mode de tracé des polygones

Le coût du rendu n'est pas celui qu'on croit...

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

```
void glPolygonMode (GLenum face, GLenum mode);
```

Défini le mode de tracé pour le(s) face(s) concernée(s).

*face* : GL\_FRONT\_AND\_BACK, GL\_FRONT, GL\_BACK

*mode* : GL\_POINT, GL\_LINE, GL\_FILL (défaut)



# Élimination des surfaces cachées

## Vision générale

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

### ■ Bas niveau :

#### ▶ Culling

Limitations : objets concaves

#### ▶ Z-Buffer

Limitations : précision des calculs (offset), arêtes communes, transparence

### ■ Algorithmique de tri :

#### ▶ Listes de priorités

Limitations : ambiguïtés, recouvrements cycliques

#### ▶ Partitionnement d'espace (ex. BSP tree)

Limitations : géométrie statique



# Suppression de (sur)faces par culling

Au niveau des polygones

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

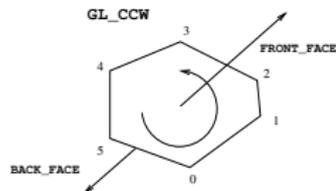
Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

Un polygone a deux faces appelées  
FRONT et BACK



```
void glFrontFace (GLenum mode);
```

Défini l'orientation des polygones.

*mode* : GL\_CCW (defaut), GL\_CW

```
void glCullFace (GLenum mode);
```

Défini le culling.

*mode* : GL\_BACK, GL\_FRONT, GL\_FRONT\_AND\_BACK

```
glEnable / glDisable (GL_CULL_FACE);
```

Demande à OpenGL de (dés)activer le culling.



# Suppression de zones par Z-buffer

Au niveau de la rasterization

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

## ■ Requête :

- ▶ **GLX** : `glXChooseVisual()` avec `GLX_DEPTH_SIZE`
- ▶ **GLUT** : `glutInitDisplayMode()` avec `GLUT_DEPTH`

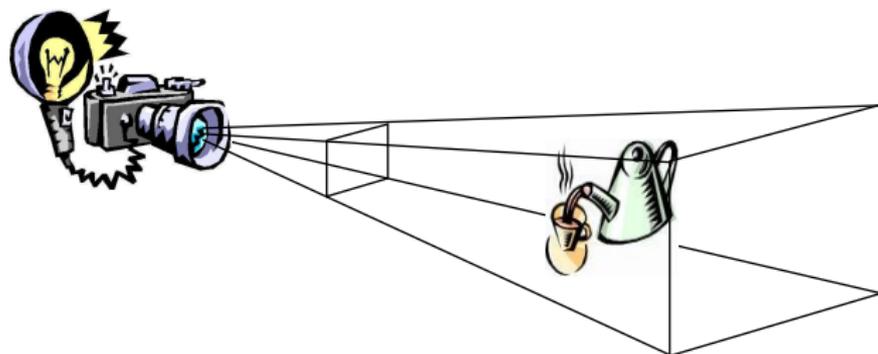
## ■ (Dés) Activation :

```
glEnable / glDisable (GL_DEPTH_TEST);
```



# Le pipeline géométrique

Vision générale



OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

- 1 Arranger la scène *modèle*
- 2 Pointer l'appareil vers la scène *vue*
- 3 Paramétrer l'objectif (grand angle, zoom. . . ) *projection*
- 4 Régler l'agrandissement et tirer la photo *viewport*



# Équivalent informatique

Bienvenue dans la matrice...

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

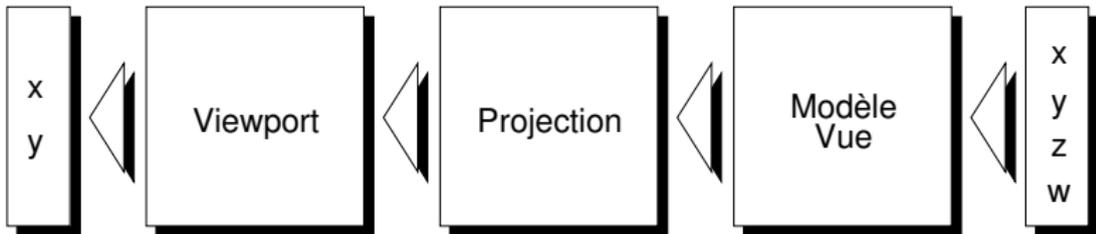
Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

Coordonnées graphiques  
sur la fenêtre



- **Remarque** : dualité modèle  $\iff$  vue
- **Implémentation** : transformations matricielles (affines)
- **Dimension 4** : problème de la translation
- **Coordonnée homogène** :  $w$  (point / vecteur)



# Mode matriciel d'OpenGL

Sur quelle matrice travaille t'on ?

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

Deux modes : **Modelview** et **Projection**

```
void glMatrixMode (GLenum mode);
```

Défini la matrice courante (affectée par les prochaines transformations).

*mode* : GL\_MODELVIEW (défaut), GL\_PROJECTION



# Préliminaires

Des choses qui changent rarement

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

```
void glClearColor (GLclampf red, GLclampf green,  
GLclampf blue, GLclampf alpha);
```

Spécifie la couleur de nettoyage (du color buffer).

```
void glClearDepth (GLclampf depth);
```

Spécifie la profondeur de nettoyage (du Z-buffer).



# Avant / Après :

À ne pas oublier !

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

## ■ Avant traçage :

```
void glClear ( GLbitfield mask);
```

Nettoie les buffers spécifiés dans le masque.

*mask* : GL\_COLOR\_BUFFER\_BIT,  
GL\_DEPTH\_BUFFER\_BIT

## ■ Après traçage :

```
void glFlush (void);
```

Force l'exécution de toutes les commandes latentes.

```
void glFinish (void);
```

Idem, et attend la fin de l'exécution.

- **Remarque** : Les commandes qui swappent les buffers en mode double flushent automatiquement.



# Commandes spécifiques

## Manipulations matricielles indirectes

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue

Projection  
Viewport

Résultat

Les fonctions suivantes définissent des matrices qui multiplient (à droite) la matrice courante :

```
void glTranslate{fd} (TYPE x, TYPE y, TYPE z);
```

Applique une translation de  $x, y, z$  sur les trois axes.

```
void glRotate{fd} (TYPE a, TYPE x, TYPE y, TYPE z);
```

Applique une rotation de  $a$  degrés autour de l'axe  $\overrightarrow{x, y, z}$

```
void glScale{fd} (TYPE x, TYPE y, TYPE z);
```

Applique une homothétie de  $x, y, z$  sur les trois axes.  
Utiliser `glScale*` avec parcimonie.



# Commandes générales

## Manipulations matricielles directes

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

```
void glLoadIdentity (void);
```

Charge l'identité dans la matrice courante (défaut).

```
void glLoadMatrix{fd} (const TYPE *mat);
```

Charge la matrice courante avec celle spécifiée.

```
TYPE mat [16] ;
```

```
M ← mat
```

```
void glMultMatrix{fd} (const TYPE *mat);
```

Multiplie la matrice courante par celle spécifiée.

```
TYPE mat [16] ;
```

```
M ← M.mat
```



# Commandes liées aux états

OpenGL dispose d'une mémoire d'états

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

```
void glGetFloatv (GLenum name, GLfloat *mat);
```

Récupère la matrice courante spécifiée.

```
name : GL_MODELVIEW_MATRIX,  
GL_PROJECTION_MATRIX
```

```
void glPushMatrix (void);
```

Ajoute une matrice sur la pile de matrices courante, initialisée comme la précédente.

```
void glPopMatrix (void);
```

Enlève une matrice de la pile de matrices courante.

- **Modelview** : pile de profondeur 32
- **Projection** : pile de profondeur 2



# Réfléchissez à l'ordre des transformations !!

Si vous ne voulez pas perdre vos cheveux

OpenGL

Didier Verna  
ENSTA  
D9-1

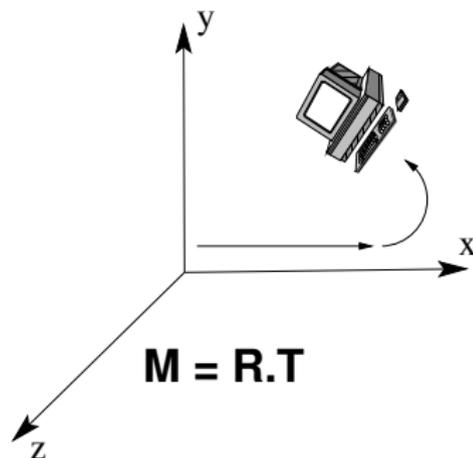
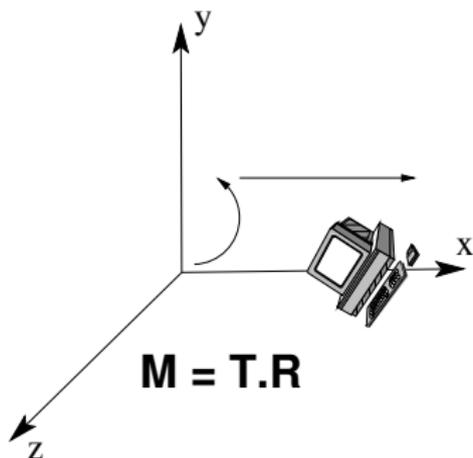
Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat



## Attention :

- Ordre des transformations inversé dans le code
- Matrices en système américain



# Placement du point de vue

C'est pareil

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue

Projection  
Viewport

Résultat

Les mêmes fonctions que les précédentes, plus :

```
void gluLookAt (GLdouble ex, GLdouble ey, GLdouble ez,
```

```
GLdouble cx, GLdouble cy, GLdouble cz,
```

```
GLdouble ux, GLdouble uy, GLdouble uz);
```

Positionne l'« oeil » au point  $E$ , oriente le point de vue dans la direction  $\overrightarrow{EC}$ , et défini le « haut » par le vecteur  $\overrightarrow{U}$ .

La matrice obtenue multiplie (à droite) la matrice courante.



# Projection en perspective

## Réalisme visuel

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

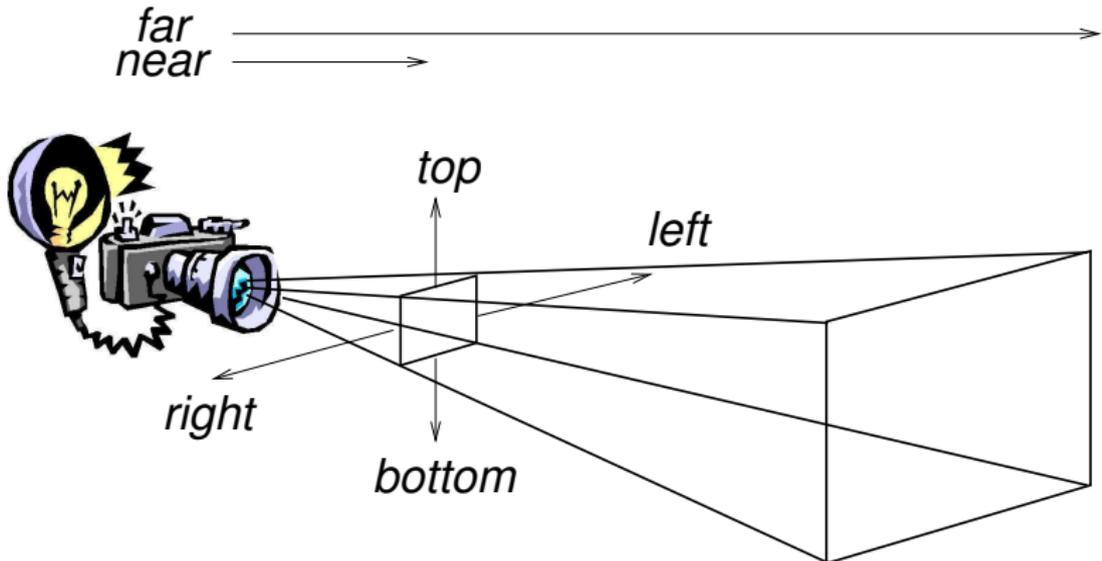
Scène

Objets  
Point de vue

Projection  
Viewport

Résultat

**void** glFrustum (GLdouble left, GLdouble right,  
GLdouble bottom, GLdouble top,  
GLdouble near, GLdouble far);





# Alternative GLU

Plus simple, plus limitée

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

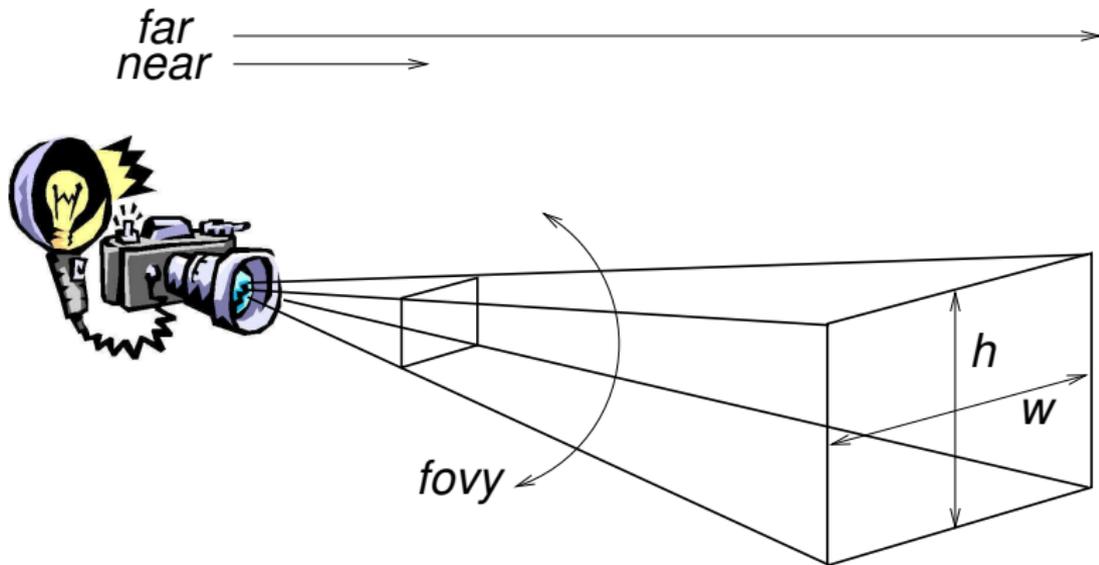
Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection

Résultat

**void** gluPerspective (GLdouble fovy, GLdouble aspect,  
GLdouble near, GLdouble far);





# Projection orthographique

Conservation des distances

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

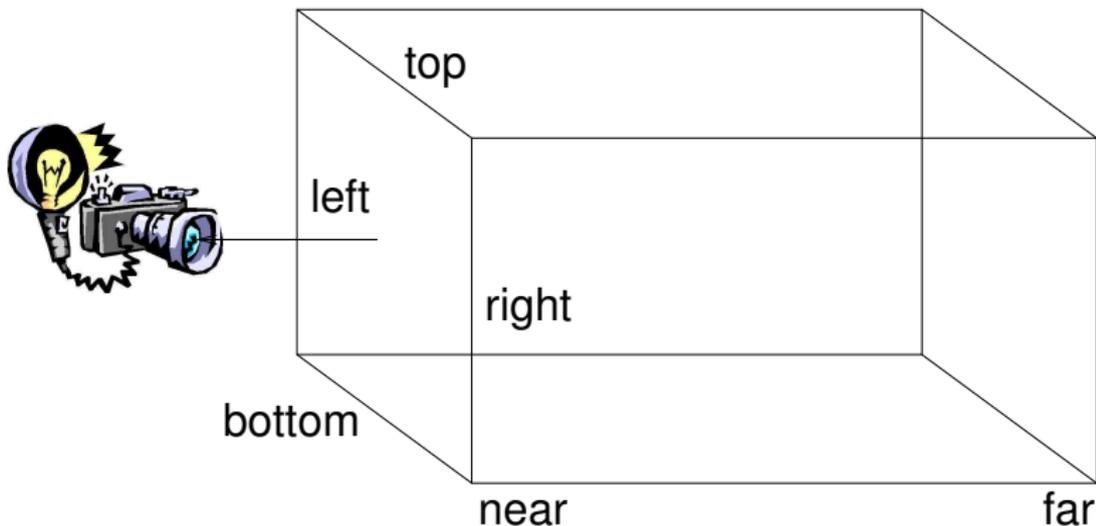
Scène

Objets  
Point de vue

Projection  
Viewport

Résultat

**void** glOrtho (GLdouble left, GLdouble right,  
GLdouble bottom, GLdouble top,  
GLdouble near, GLdouble far);





# Transformation de Viewport

Où tracer, au final ?

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

```
void glViewport (GLint x, GLint y, GLsizei w, GLsizei h);
```

## Fenêtre OpenGL

Zone de tracé

h

y

x

w



# Résultat du TP

Ce que vous devriez obtenir...

OpenGL

Didier Verna  
ENSTA  
D9-1

Géométrie

Primitives  
Surfaces cachées

Scène

Objets  
Point de vue  
Projection  
Viewport

Résultat

