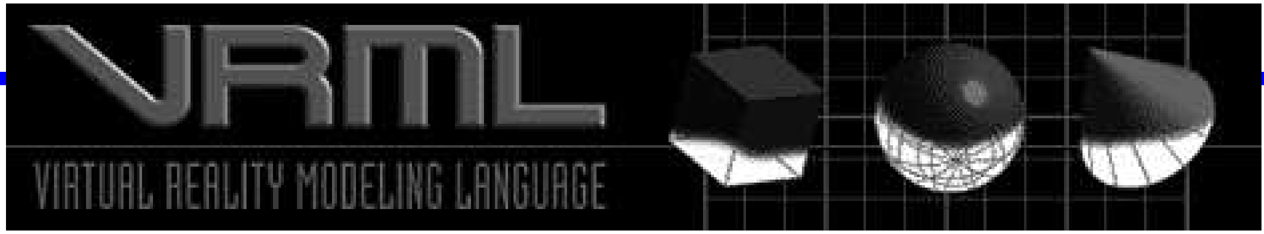
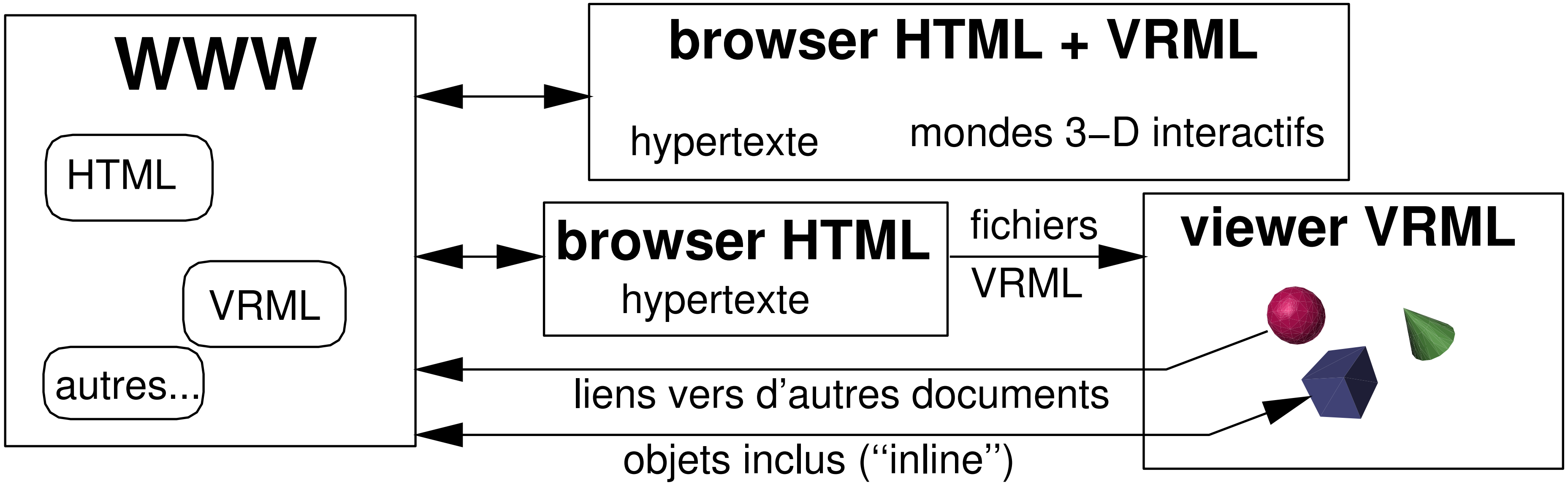
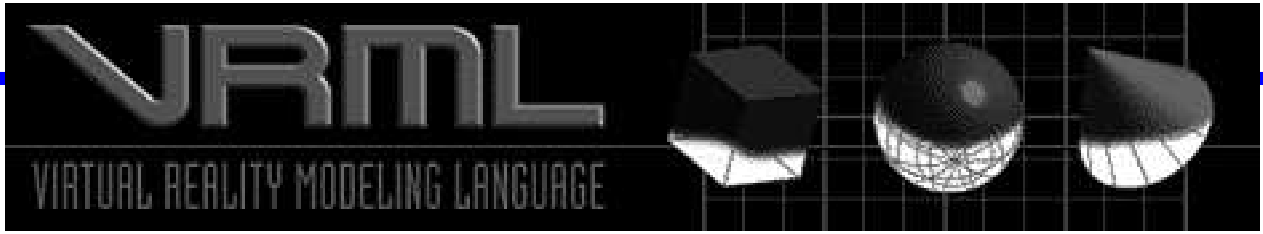



**VRML = Langage de description de mondes virtuels interactifs:**

– multi–participants

– reliés par réseau (Internet + WWW)






**VRML 1.0 contient :**

– objets 3–D (formes, positions, matériaux) ;

– liens (“anchor”) vers d’autres documents ou mondes ;

– objets 3–D inclus (“inline”).

**Prochaines versions (VRML 2.0...) :**

– interactions avec l’utilisateur et le monde (comportement) ;

– mouvement, animation ;

– multi–utilisateurs.

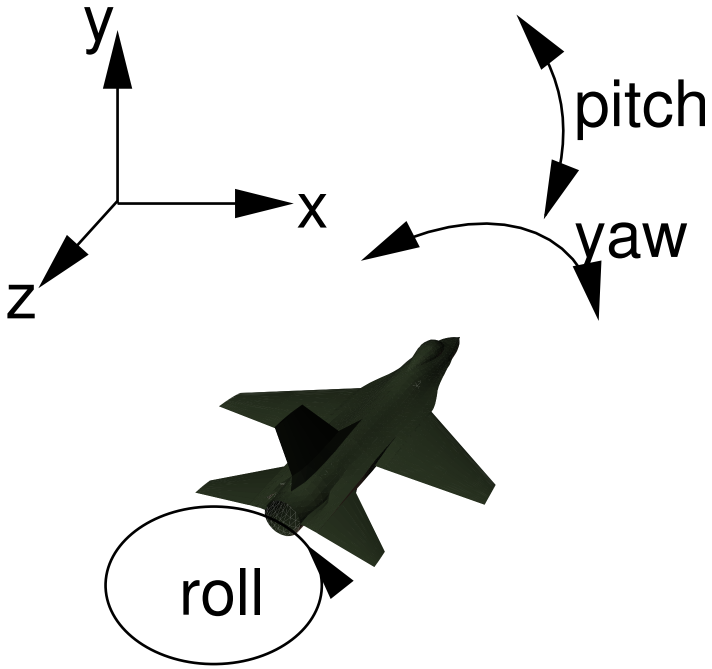
➪ nécessité d’utiliser un langage de script (tcl, Java) pour décrire ces nouvelles notions.

**Informations sur VRML :**

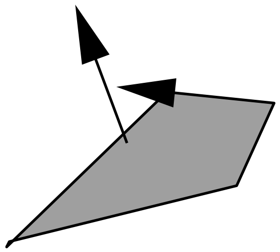
**http://vrwml.wired.org/** et **http://www.sdsc.edu/vrml/**

**http://www.hyperreal.com/~mpesce/book/**

## Un peu de géométrie 3–D et de graphisme...



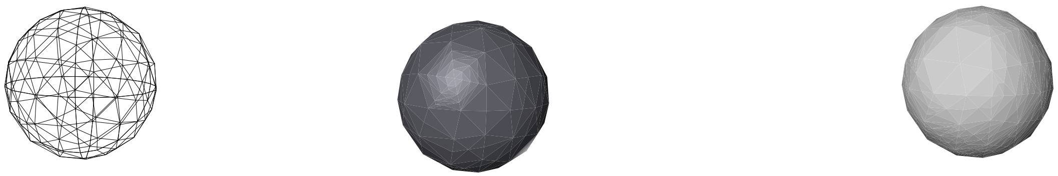

pitch = tangage
roll = roulis
yaw = lacet



polygones orientés par le ‘‘tire–bouchon’’

**Types d’éclairages : lumière ponctuelle, parallèle, ou spot.**

**Rendus : filaire, plat, gouraud, phong, texturé**



## Le format de fichier VRML 1.0

– Scène représentée par un graphe ;

– L'ordre des noeuds est important (un noeud peut affecter les noeuds suivants dans le graphe, par exemple `Rotation` ou `Material`) ;

– Le noeud `Separator` permet de limiter l'effet de telles propriétés à un sous–arbre ;

– Un noeud contient :

  – le type d'objet (cube, sphère, texture, transformation) ;

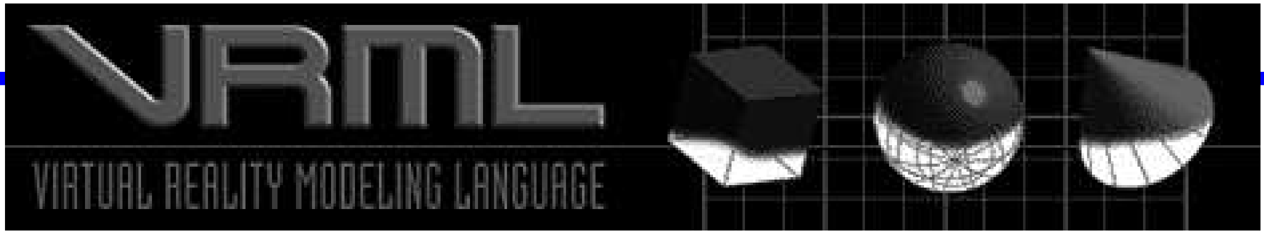
  – ses paramètres (dimensions...) ;

  – son nom (facultatif) ;

  – les noeuds fils (pour les noeuds "groupe").

– Syntaxe d'un noeud :

```
DEF nomdelobjet typedobjet { champs noeudsfils }
```




– En–tête de fichier :

```
#VRML V1.0 ascii
```

– Commentaires commencent par #

– Exemple simple :

```
#VRML V1.0 ascii
# une sphere jaune
Separator {                                  # le noeud Separator groupe ce qu'il contient ensemble
        Material {                           # rouge+vert=jaune
                diffuseColor 1 1 0           # diffuseColor = couleur de base
        }
        Sphere {
                radius 10                    # rayon de 10 (en metres !)
        }
}
```

– Dans un groupe, tous les noeuds fils sont dans le même repère

➪ utilisation des transformations (translations, rotations, facteurs d'echelle)

## Les transformations

– étant donné un point **V** = (xV, yV, zV),

```
Translation {        translation xT yT zT }
Rotation { rotation xR yR zR aR }
Scale { scaleFactor xS yS zS }
Coordinate3 { point xV yV zV } PointSet { numPoints 1 }
```

transforme **V** en **V'=T.R.S.V** (points = vecteurs colonnes)

– Ces transformations se font **dans le repère de l'objet** (local).

penser :

– je me translate ;

– je tourne autour de l'axe qui se trouve là par rapport à moi ;

– je me dilate sur chacun des axes de coordonnées ;

– je me dessine.

## Le noeud Transform

```
Transform {
        translation      xT1 yT1 zT1
        rotation         xR1 yR1 zR1 aR1    # direction + angle (rad)
        scaleFactor      xS yS zS           # facteur d'echelle
        scaleOrientation xR2 yR2 zR2 aR2
        center           xT2 yT2 zT2
}
```

équivalent à

```
Translation { translation T1 }
Translation { translation T2 }
Rotation { rotation R1 }
Rotation { rotation R2 }
Scale { scaleFactor S }
Rotation { rotation –R2 }
Translation { translation –T2 }
```

## Définition/Instanciation d'objets : DEF/USE

– USE = inserer le dernier objet parsé avec ce nom défini par DEF:

```
#VRML 1.0 ascii
Separator {
        DEF Joe Sphere { }          # Sphere de rayon 1
        Translation { translation 2 0 0 }
        Separator {
                DEF Joe Sphere { radius .2 }
        }
        Translation { translation 2 0 0 }
        USE Joe                     # Sphere de rayon 0.2
}
```

## Liens : le noeud WWWAnchor

– C’est un noeud de groupe, tous ses fils sont reliés à l’URL du noeud

– Syntaxe :

```
WWWAnchor {
     name "http://www.essi.fr/"
     description "Le serveur de l’ESSI"
     Sphere {  # un fils
          radius 1
     }
}
```

## Inclusion de fichier VRML : WWWInline

–inclusion d'un fichier, donné par son URL, dans le contexte actuel

–la boîte englobante (bounding box) permet au viewer :

- –de ne charger le fichier que lorsqu'il arrive dans le champ
- –de dessiner la boîte pendant le chargement de l'objet

–Exemple :

```
WWWInline {
        name
"http://hyperreal.com/~mpesce/book/examples/second.wrl"
        bboxSize 10 10 10    # taille de l'objet
        bboxCenter 0 0 0     # L'objet est centré à l'origine
}
```

## Représentation multi–échelles : le noeud LOD

– Plusieurs représentations d'un objet, à différents niveaux de détail (LOD = Level Of Detail) ;

– Plus on est proche, plus l'objet visualisé doit contenir de détails ;

– Syntaxe :

```
LOD {
      range [  10, 100, 200 ]   # distances
      center 0 0 0      # point de ref. pour calculer la distance
      fils affiché si (dist < 10)
      fils affiché si (10 < dist < 100)
      fils affiché si (100 < dist < 200)
      fils affiché si (200 < dist)
}
```

– Mettre tous les fils dans des Separator par mesure de sécurité

## Caméras et points de vue

PerspectiveCamera et OrthographicCamera

```
PerspectiveCamera {
        position x y z                # centre de la camera
        orientation xv yv zv av       # dir. de visee et torsion
        focalDistance 250             # distance du point fixé
        heightAngle 0.7               # angle de champ (rad)
}
```

Attention : une caméra ne voit que les fils du noeud dans lequel elle est définie !

## Visite guidée et indications au browser : Info, Switch et DEF

```
#VRML V1.0 ascii
DEF Title Info { string         "Maison" } # titre
DEF SceneInfo Info { string "ma maison,
scène créée le 18/02/96" } # info sur le fichier
DEF Viewer Info { string "walk" }   # utiliser le walk viewer
DEF ViewerSpeed Info { string      "10" } # vitesse de deplacement
DEF Cameras Switch {
        whichChild 1            # deuxieme camera par defaut
        DEF VueGlobale PerspectiveCamera { # camera 0
                position ...
        }
        DEF Entree PerspectiveCamera { ... } # camera 1
        DEF Jardin PerspectiveCamera { ... } # camera 2
}
```

## Écriture de mondes VRML

– S’inspirer d’exemples simples, utiliser WWWInline et LOD.

– Banques d’objets disponibles sur WWW

– Nombreux traducteurs disponibles

– Création automatique à partir de scripts CGI

– La librairie QvLib pour parser les fichiers VRML ➪ objets C++ (surtout utile pour faiure un browser)

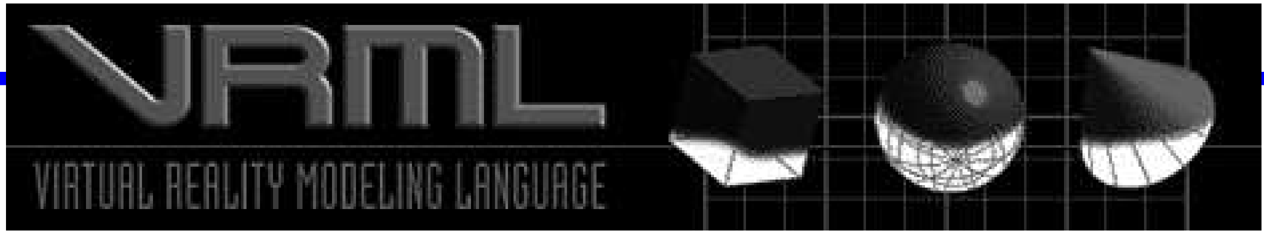
## Le Browser

– Nombreuses implémentations dans le monde PC

– Sur Unix : Webspace (SGI), VRWeb, et Liquid Reality (Java/Solaris)

– Interface utilisateur souvent compliquée d’utilisation (pas de notion d’échelle de vitesse ou de distance en VRML 1.0, souvent pas de détection de collision, souris inadaptée à la 3D)

– Astuces spécifiques à un browser dans les noeuds Info : http://webspace.sgi.com/Repository/Palladium/Palladium.wrl.gz




## VRWeb

– pas de WWWInline !

options de navigation

options d'affichage

afficher les liens

tourner la tête (h/b/d/g)

marcher (av/ar/d/g)

translation (h/b/d/g)

aller vers :

viser = bouton G

avance = bouton M

recule = bouton D