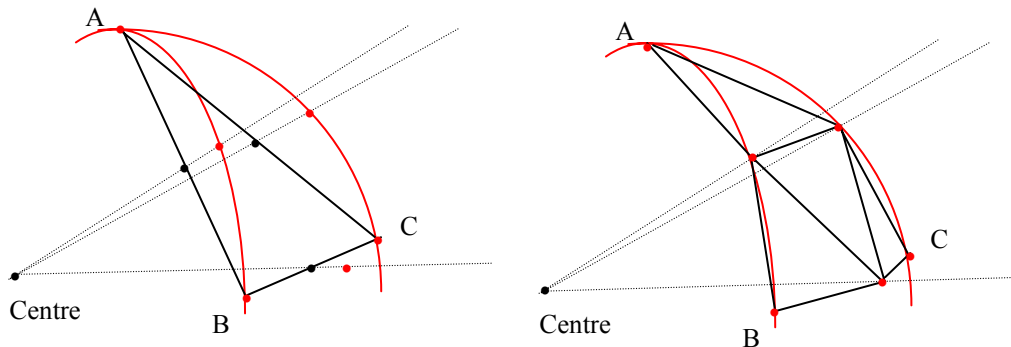


# TD LO12

## 1. Modélisation

### 1) « Facettisation » d'une sphère

Principe : On part d'un tétraèdre régulier dont les sommets appartiennent à une sphère de rayon 1. Pour chaque face de ce tétraèdre, on construit comme le montre la figure ci-dessous, 4 nouveaux triangles dont les sommets appartiennent à la sphère

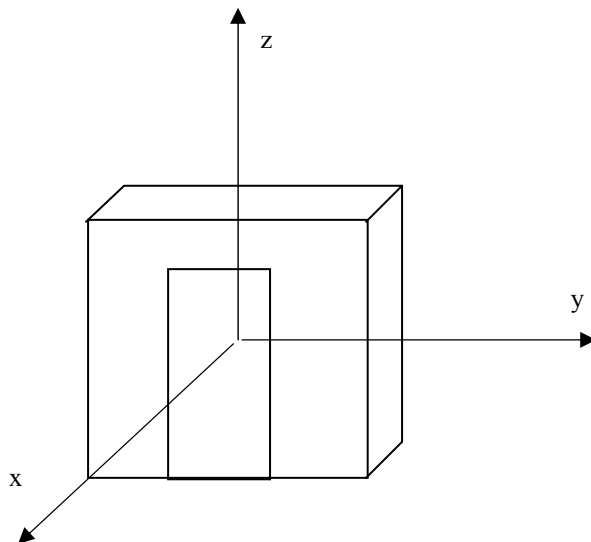


- Initialiser les coordonnées des sommets du tétraèdre initial. La sphère est centrée en  $(0,0,0)$
- Ecrire la fonction `facettisationSphere(n)` qui affiche une sphère facettisée par  $n$  subdivisions du tétraèdre initial.  
Pour cela on supposera connue la fonction `afficheTriangle(A,B,C)`

### 2) Description statique d'une scène

Introduction aux scènes utilisées en TP.

Ecrire un fichier de données contenant la description de la scène suivante (1 objet = cube avec porte).  
Cube de taille 2, centré en  $(0,0,0)$  avec une porte de dimension  $0,6 \times 1,6$  comme indiqué ci-dessous.  
Le format du fichier est décrit au verso de la feuille.



Le fichier de données doit respecter le format suivant :

nombre de sommets (1 ligne)  
nombre de matières (1 ligne)  
nombre de faces (1 ligne)  
nombre d'objets (1 ligne)  
nombre de sources (1 ligne)  
coordonnées x y z des sommets (1 point par ligne, valeurs séparées par une tabulation)  
description des matières (6 lignes par matières) : **donnée en TP**  
description des faces (3 lignes/face)  
    nombre de sommets pour la face  
    indices des sommets séparés par une tabulation.  
    indice de la matière  
description des objets (6 lignes / objet)  
    nombre de faces  
    indices des faces séparés par une tabulation.  
    Matrice de transformation liée à l'objet (4 lignes de 4 réels séparés par une tabulation)  
description des sources (5 lignes/source) : **donnée en TP**

Remarque :

les nombres et indices sont des entiers (int)

les coordonnées, angles et coefficients RGB (matières) sont des réels (double)

Les commentaires sont précédés du symbole #

## 2. Dessin 2D

1) Algorithme de Bresenham.

Rappeler les étapes du tracé de cercle par l'algorithme de Bresenham .  
Retrouver les instructions de l'algorithme.

2) Liste de sommets

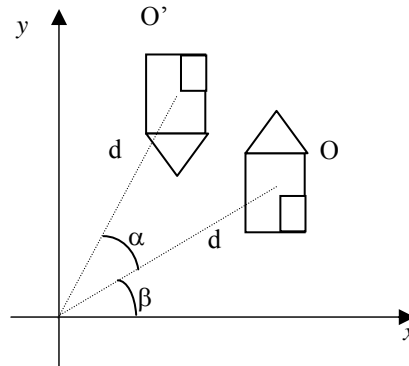
Soit un polygone convexe décrit par la liste (non ordonnée) de sommets  $P_i$  ( $i > 3$ ). Ecrire l'algorithme qui affiche les contours du polygone. On suppose connue la fonction `afficheSegment (P1, P2)`.

3) Courbes de Bézier

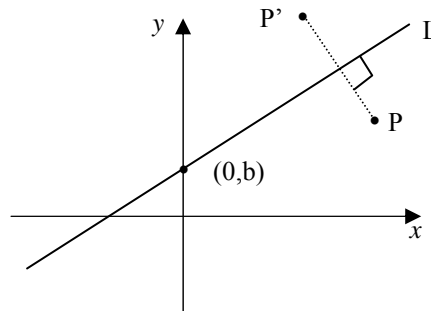
- a) Calculer le polynôme et tracer la courbe de Bézier pour les points de contrôle  $P_0(1,1)$   $P_1(8,0)$   $P_2(2,-5)$
- b) Même question pour les points de contrôle  $P_0(1,1)$   $P_1(8,0)$   $P_2(8,0)$   $P_3(2,-5)$  .
- c) Même question pour les points de contrôle  $P_0(1,1)$   $P_1(8,0)$   $P_2(8,0)$   $P_3(8,0)$   $P_4(2,-5)$

### 3. Transformations

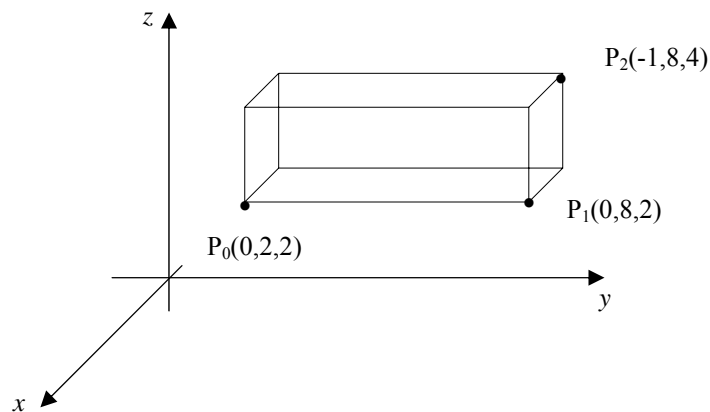
- 1) Quelle est la transformation qui permet de transformer l'objet O en O' ?



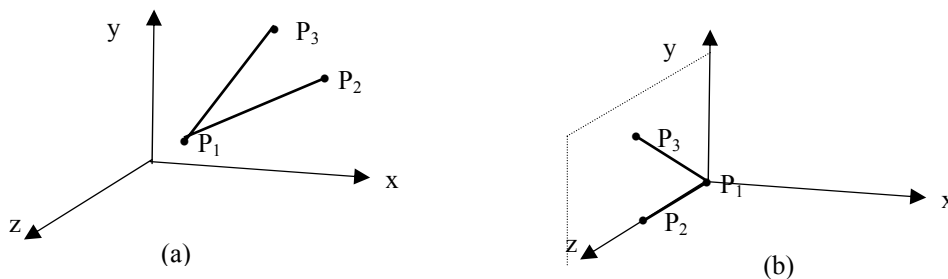
- 2) Décrire la transformation qui donne le symétrique d'un objet par rapport à une droite de pente m coupant l'axe des y en (0,b).



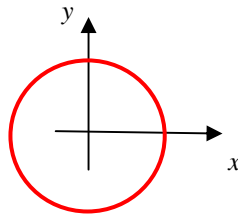
- 3) 1-Comment se transforme le parallélépipède de la figure 2 par le changement d'échelle  $E_{2,3,4}$   
 2- Comment faire pour que le centre de ce parallélépipède soit toujours au même endroit ?



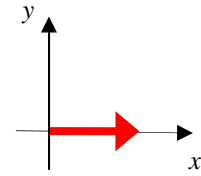
- 4) Trouver la transformation permettant de se ramener de (a) à (b)



5) Soit les objets suivants :



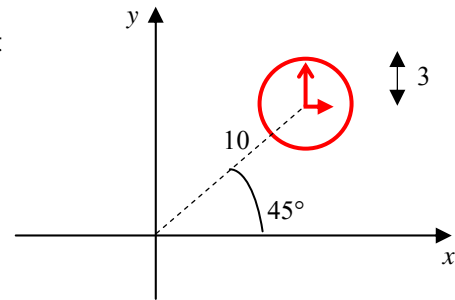
Objet 1 : Cercle de rayon 1



Objet2 : flèche de taille 1

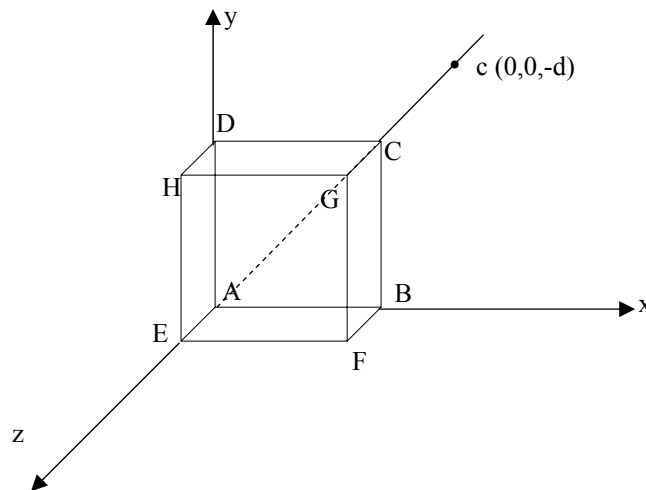
1) Donner les transformations à appliquer à chaque élément de la scène pour afficher le dessin ci-contre.

2) Proposer le code OpenGL correspondant



6) Projections

On projette le cube unité de la figure ci-dessous sur le plan Oxy. Dessiner l'image projetée par la transformation perspective standard avec  $d=1$  puis  $d=10$ ,  $d$  étant la distance du centre de projection au plan de projection



7) Soit la projection perspective donnée par les calculs suivants :

$$Pe \begin{pmatrix} xe \\ ye \\ d \\ 1 \end{pmatrix} \quad \text{point projeté sur le plan de projection}$$

$$Po \begin{pmatrix} xo \\ yo \\ zo \\ 1 \end{pmatrix} \quad \text{point exprimé dans l'espace de l'observateur}$$

$d$  distance du plan de projection à l'observateur

$$xe = \frac{dxo}{zo}$$

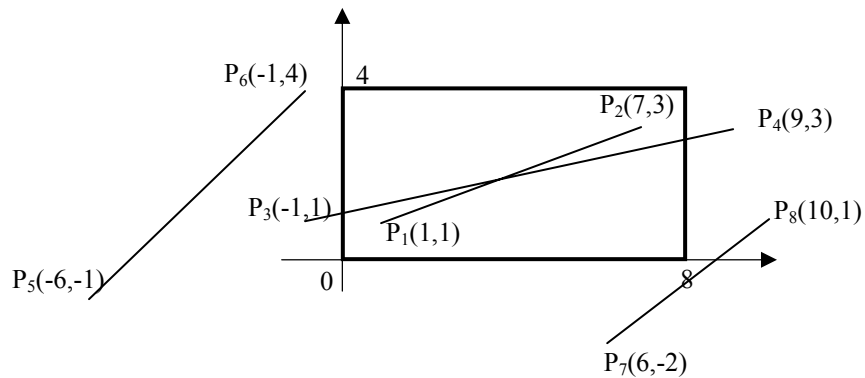
$$ye = \frac{dyo}{zo}$$

- 1) Trouver la matrice de projection homogène  $M_p$  [4x4] correspondant à ces formules telle que  $Pe = M_p Po$
- 2) Quelles hypothèses fait-on lorsqu'on applique ces formules ?

## 4. Fenêtrage

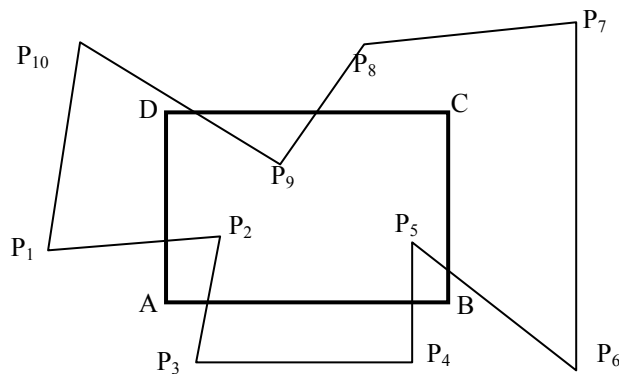
### 1) Fenêtrage de segments.

- Appliquer l'algorithme de Cohen-Sutherland pour le fenêtrage des segments de la figure suivante.
- idem avec l'algorithme de Cyrus-Beck.



### 2) Algorithme de Sutherland-Hodgman

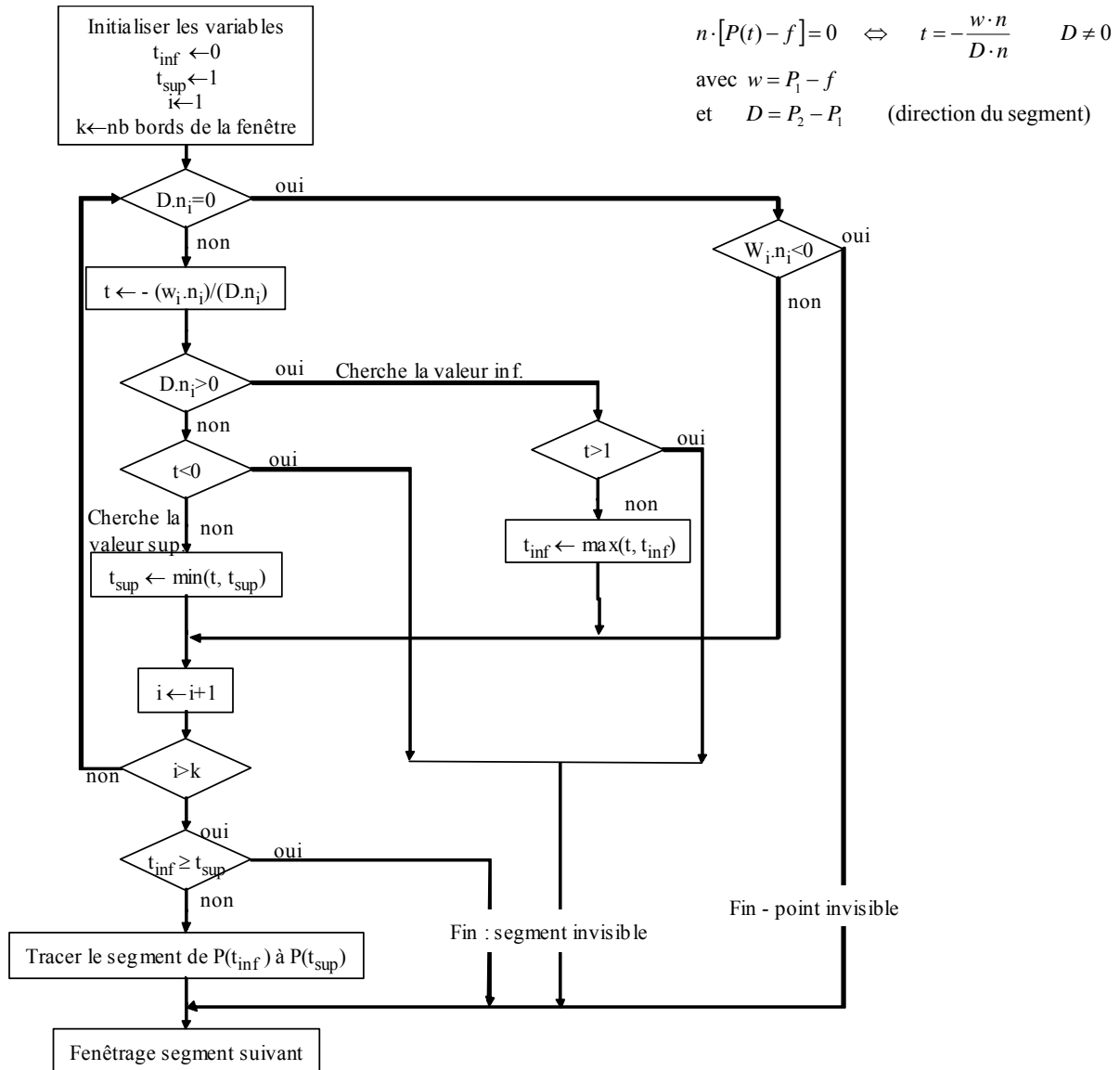
- Fenêtrer le polygone  $P_1, \dots, P_{10}$  de la figure ci-dessous avec la fenêtre ABCD à l'aide de l'algorithme de Sutherland-Hodgman



- Comment modifier l'algorithme de Sutherland-Hodgman pour qu'il effectue un masquage et non un fenêtrage ?



### Algorithme de Cyrus-Beck



### Algorithme de Cohen-Sutherland

1. Affecter un code à 4 bits à chaque extrémité du segment
  - bit 1 = 1 : l'extrémité est au dessus de la fenêtre
  - bit 2 = 1 : l'extrémité est en dessous de la fenêtre
  - bit 3 = 1 : l'extrémité est à droite de la fenêtre
  - bit 4 = 1 : l'extrémité est à gauche de la fenêtre
2. Classifier le segment

si $code1=0000$ et $code2=0000$	le segment est visible totalement
sinon si $(code1 \text{ et } code2) \neq 0000$	il est invisible
sinon	le segment est divisé
3. Examiner le segment restant :
  - Calculer l'intersection du segment avec la droite du bord concerné et remplacer l'extrémité par cette intersection
  - calculer son code d'extrémité
  - aller en 2 avec le nouveau segment

## 5. Elimination des parties cachées

### 1) Z-buffer.

Le z-buffer avec une vision en perspective pose un problème (inverse !) : on doit connaître la profondeur  $z$  du point  $P$  associé au pixel  $(x_e, y_e)$  lors d'un tracé de segment.

Soit segment  $[P_1P_2]$  dans le repère obs.  $P_1 \begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} \quad P_2 \begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix}$

Soit le segment 2D  $[P_1'P_2']$  obtenu par projection perspective du segment  $[P_1P_2]$  sur un écran situé à une distance  $d$  du centre de projection (centre du repère) et perpendiculaire à l'axe  $z$ .

Soit  $P \in [P_1P_2]$  et  $P'$  son projeté.

Montrer que si  $P' = P_1' + \rho(P_2' - P_1')$  alors

$$z = \frac{z_1 \cdot z_2}{z_2 - \rho(z_2 - z_1)}$$

### 2) Algorithme de remplissage : cotés actifs et z-buffer

Appliquer l'algorithme de remplissage des cotés actifs avec z-buffer à la scène composée des 2 polygones R et T. L'observateur est à l'infini sur l'axe des  $z$  positifs.

R a pour sommets :

$$P_1 \begin{pmatrix} 10 \\ 5 \\ 10 \end{pmatrix} \quad P_2 \begin{pmatrix} 10 \\ 25 \\ 10 \end{pmatrix} \quad P_3 \begin{pmatrix} 25 \\ 25 \\ 10 \end{pmatrix} \quad P_4 \begin{pmatrix} 25 \\ 5 \\ 10 \end{pmatrix}$$

T a pour sommets :

$$P_5 \begin{pmatrix} 15 \\ 15 \\ 15 \end{pmatrix} \quad P_6 \begin{pmatrix} 25 \\ 25 \\ 5 \end{pmatrix} \quad P_7 \begin{pmatrix} 30 \\ 10 \\ 5 \end{pmatrix}$$

## Algorithme des cotés actifs et parties cachées

- v Gestion de 3 tables (initialisation)
  - TP : table des polygones, chaque polygone est décrit par
    - Id : identificateur du polygone
    - Eq. du plan : coefficients ABCD
    - couleur: couleur de remplissage
    - flag : polygone actif ou non
  - TA: table des arêtes, chaque arête est décrite par
    - x initialisé pour  $y_{\min}+1/2$
    - $y_{\max}$ : hauteur max de l'arête
    - $\Delta x$  : inverse de la pente
    - Id : identificateur du polygone
  - TAA : table des arêtes actives
    - y : ligne de balayage
    - Initialisée avec les arêtes dont le  $y_{\min} = y$*
- v Générer le tracé (ligne y)
  - On procède comme pour l'algorithme de cotés actifs mais on détecte les "mauvaises paires" en modifiant le flag (booléen) de chaque polygone à chaque fois qu'on traite un coté de celui ci :
  - Si le flag de 2 polygones sont à 1 (vrai) simultanément, il faut calculer les profondeurs et on allume le pixel avec la couleur du polygone correspondant au  $z_{\min}$
  - Sinon on traite les paires sans calculer la profondeur et on allume le pixel avec la couleur du polygone correspondant au coté actif.
- v Propager à la ligne suivante (ligne y+1)
  - Si  $y+1 > y_{\min}$ , l'arête n'est pas propagée (elle n'est plus active)
  - Si non,
    - modifier x en le remplaçant par  $x + \Delta x$
    - ajouter l'arête à la liste de la ligne (y+1)
  - Générer le tracé en (y+1)

## 6. Eclairage

### 1) Rendu : réflexion spéculaire

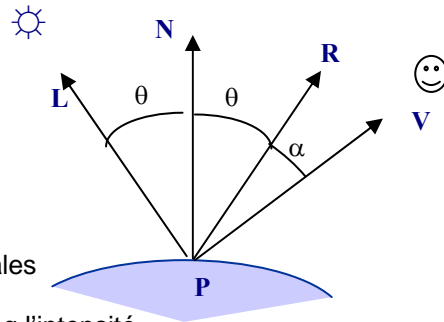
- Dans le cas de la réflexion spéculaire, déterminer le vecteur R à partir de L et N.
- Que se passe-t-il si la source est placée à l'infini sur l'axe z positifs ?
- D'une manière générale, que signifie une source située à l'infini ?

### 2) Calcul d'intensité avec le modèle de Phong

Soient les vecteurs  $\vec{N} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$   $\vec{L} \begin{pmatrix} -1 \\ 2 \\ -1 \end{pmatrix}$   $\vec{V} \begin{pmatrix} 1 \\ 3/2 \\ 1/2 \end{pmatrix}$

On suppose l'objet éclairé par les lumières ambiantes et locales telles que  $I_a=0.1$   $I_p=1$   $k_a=0.15$   $k_d=0.15$   $k_s=0.8$   $n=5$

Déterminer à l'aide du modèle d'illumination de Phong l'intensité au point considéré.



### 3) Problèmes liés à l'interpolation

Dans les méthodes d'ombrage de Gouraud et de Phong, on interpole respectivement les intensités et les normales pour ombrer un point P situé à l'intérieur d'un polygone de la scène.

- Montrer que dans le cas d'un polygone à 4 cotés l'ombrage de Phong et de Gouraud est sujet à des problèmes de discontinuités (en animation).
- Afin de lisser le maillage polygonal, on peut définir en chaque sommet une normale correspondant à la moyenne des normales des facettes auxquelles appartient le sommet. quels problèmes peut-on rencontrer avec cette méthode ?

### 4) Calcul d'intensité par interpolation de Gouraud et Phong

Soit l'objet décrit dans la figure 2 et dont les équations de plan des facettes sont les suivantes :

plan 1 :  $z=2$

plan 2 :  $-x+1.732y+7.5z-17=0$

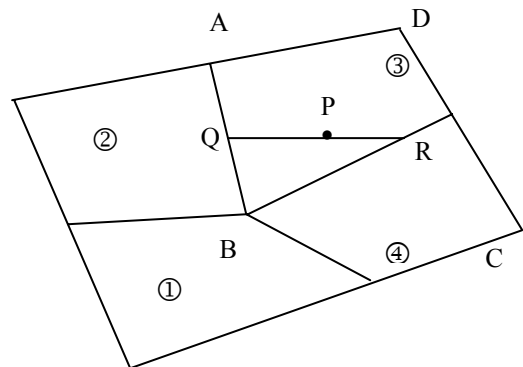
plan 3 :  $-2.25x+3.897y+10z-24.5=0$

plan 4 :  $5.5z-11=0$

Cet objet est éclairé par une source située à l'infini sur l'axe des z positif

On suppose  $\frac{AQ}{AB} = 0.4$   $\frac{BR}{BC} = 0.7$   $\frac{QP}{QR} = 0.5$   $\vec{V} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$

$I_a=0.1$   $I_p=1$   $k_a=0.15$   $k_d=0.15$   $k_s=0.8$   $n=2$



Calculer l'intensité au point P avec les 3 méthodes : ombrage constant, ombrage de Gouraud et ombrage de Phong

### 5) Modèles utilisés par OpenGL

## 7 et 8 TD Machine

Ecrire un programme qui réalise la facettisation d'une sphère par subdivision récursive des triangles composant un tétraèdre (voir TD 1) et qui propose différents modes d'affichage :

mode = 0	Filaire
mode = 1	Flat
mode = 2	Gouraud sans lissage
mode = 3	Gouraud avec lissage

Vous pouvez récupérer et compléter le fichier *eclairageTD\_etu.c* sur le site de l'UV

Vous pouvez aussi utiliser le code déjà fait en TP pour éventuellement animer la sphère (déplacements).

## 9. Textures

### 1) Mapping direct et mapping inverse

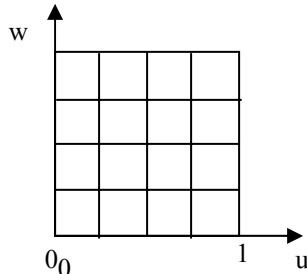


figure 1

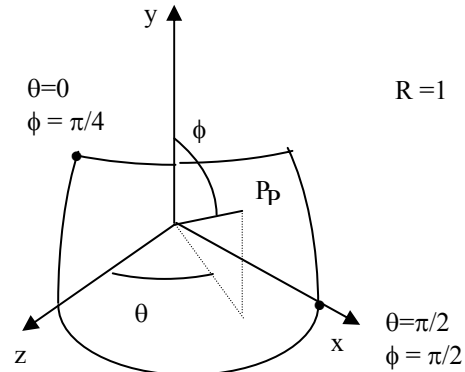
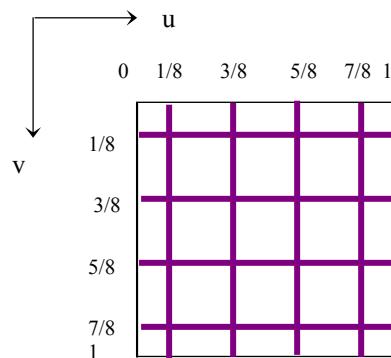


figure 2

- Donner les formules qui permettent d'appliquer la texture donnée par la figure 1 sur la surface décrite par la figure 2 (1/8 de demi sphère). (mapping direct)
- La scène est observée après rotation de  $-45^\circ$  /axe Y puis  $35^\circ$  /axe x et projetée sur un écran de taille 32X32 pixels par une projection parallèle orthogonale. Calculer l'intensité du pixel de coordonnées (21, 15) sachant que la fenêtre dans l'espace objet qui correspond à l'écran est  $x' \in [-1, +1]$  et  $y' \in [-1, +1]$ . (mapping inverse)  
On supposera la texture définie par un bitmap 64x64 pixels.

### 2) Mapping par objet intermédiaire

On souhaite appliquer la texture définie par le bitmap suivant sur un cube de taille 2 centré en (0,0,0)



- Décrire le résultat du mapping de la texture sur le cube en appliquant un mapping sphérique centré.
- Décrire le résultat du mapping de cette même texture sur le même cube en appliquant un mapping cylindrique 'centré'
  - avec un cylindre de hauteur égale à 2
  - avec un cylindre de hauteur égale à 8/3

## 10 . Suivi de rayon et animation

### 1) Calcul d'intensité par suivi de rayon

Calculer l'intensité donnée par un suivi de rayon à partir de la scène suivante (2D simplifiée)

L'observateur est à l'infini sur l'axe des z positifs en  $x=5$ .

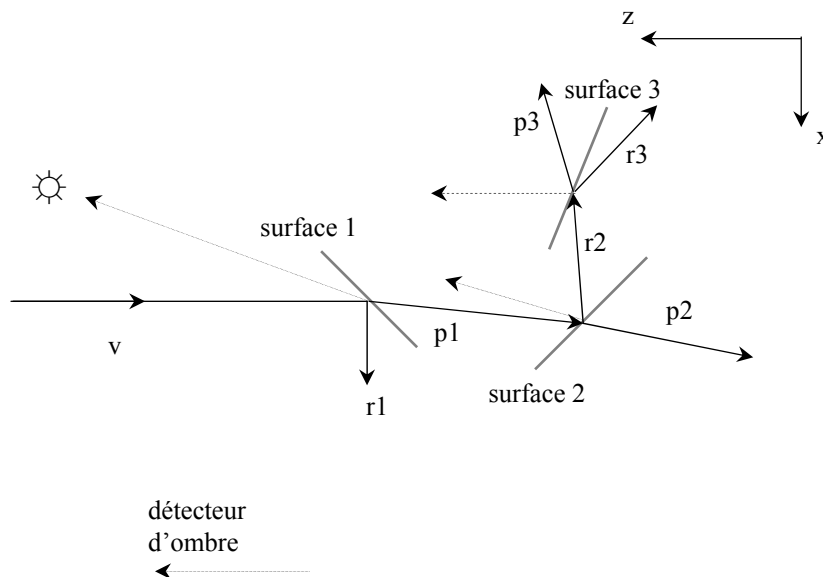
Une source lumineuse ponctuelle est placée en  $x=3, z=10$ . Son intensité  $I_l=1$ . L'intensité de la lumière ambiante  $I_a=1$

3 surfaces dans la scène :

Surface 1 :  $x+z-12.5=0$        $4 \leq x \leq 6$        $k_{a1} = 0.15$   $k_{d1} = 0.15$   $k_{s1} = 0.8$   $k_{t1} = 0.5$   $k_{\eta 1} = 1/1.1$   $n_1=50$

Surface 2 :  $x-z-2=0$        $4 \leq x \leq 6$        $k_{a2} = 0.15$   $k_{d2} = 0.15$   $k_{s2} = 0.8$   $k_{t2} = 0.5$   $k_{\eta 2} = 1.1$   $n_2=50$

Surface 3 :  $x-3z+9=0$        $1 \leq x \leq 3$        $k_{a3} = 0.15$   $k_{d3} = 0.15$   $k_{s3} = 0.8$   $k_{t3} = 0$   $k_{\eta 3} = 1.1$   $n_3=50$



### 2) Cube

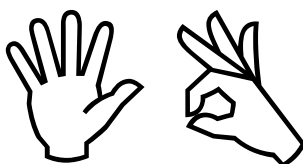
Soit un carré dont les coins sont en (0,0) et en (1,1)

- Décrire l'effet de l'animation par key frame afin que la position finale des 2 coins soit en (1,1) et (0,0) (en fait équivalent à une rotation de 180°).
- Proposer une autre méthode pour réaliser effectivement une rotation du carré de 180°

### 3) Langage gestuel

On veut réaliser l'animation d'une main afin d'illustrer une méthode d'apprentissage d'un langage gestuel par exemple.

- Proposer un squelette permettant de modéliser la main et ses articulations.
- Donner une méthode permettant d'animer la main en la faisant passer de la configuration (1) à la configuration (2)
- Donner une méthode permettant d'animer la main de la configuration (1) à un poing fermé.



(1)

(2)