3. Les bases du dessin 2D

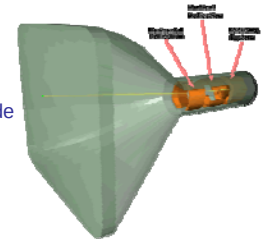
- **Mise en évidence de :**
  - Simplifications de calcul pour augmenter la rapidité
  - Echantillonnage lié à l'image : pixel
  - Contraintes liées à la technologie

3.1 Les visus à balayage de trame

- **écran : grille de points lumineux (pixels)**
- **contraintes liées à la perception humaine et à l'électronique**

continuité des images : 24 images/seconde (30 pour la vidéo)

taux de rafraîchissement : 48 images/seconde (60 vidéo)



Les visus à balayage de trame

- **entrelaçage : 1 image est affichée en 2 fois**
  lignes impaires puis paires

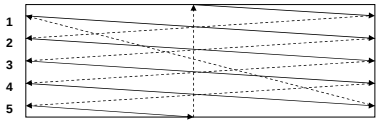


Schéma d 'un motif de balayage à 5 lignes entrelacées

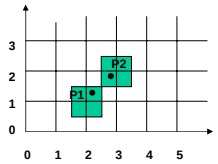
- **Quelques chiffres**

image à 525 lignes :63,5 microsecondes/ligne

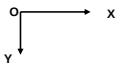
512 pixels/lignes : 103 nanosecondes/pixel

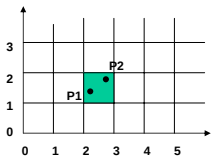
Le pixel

- **pixel (picture element)**
  - carré de taille 1
  - allumé ou éteint
  - intensité : niveaux de gris ou couleurs
- **tracé de point**
  le pixel est déterminé par des coordonnées entières
- **le repère**
  celui du balayage de trame
  les algorithmes sont illustrés avec un repère « classique »
- **affichage du point P(x,y)**
  - P1 (2.2, 1.3)
  - P2 (2.8, 1.9)



pixel (round(x), round(y))





pixel (ent(x), ent(y))

3.2 Le tracé de segment

- **Segment de droite défini par ses 2 extrémités**
  P1 (x1, y1)
  P2 (x2, y2)

- **Equation de la droite**

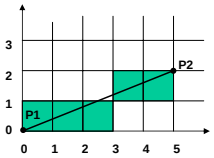
$$y = mx + b$$

$$m = \frac{y_2 - y_1}{x_2 - x_1} \qquad b = y_1 - m\,x_1$$

- **un algorithme simple**

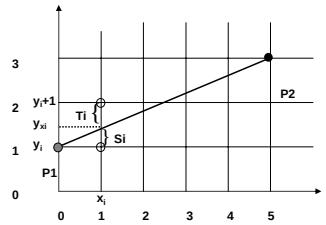
```
for (x=x1, x<x2, x++)
    { y= mx + b;
     affichePixel(x,y);
    }
```

- **effet de crénelage (aliasing)**



Algorithme de Bresenham (segment)

- **nombres entiers**
- **opérations simples :**
  - additions et soustractions
  - multiplications par 2



- **variable de décision**

T_i : distance entre $y_{x\,i}$ et le pixel au dessus
S_i : distance entre $y_{x\,i}$ et le pixel au dessous
$d_i = S_i - T_i$, variable de décision
si $d_i < 0$ pixel au dessous
si $d_i >= 0$ pixel au dessus

Algorithme de Bresenham (segment)

- **calcul des différentes valeurs de d_i:**

init : $d_1 = 2\,dy - dx$
$dy = y_2 - y_1$
$dx = x_2 - x_1$

algo : si $d_i >= 0$

alors { $x_{i+1} = x_i + 1$
$y_{i+1} = y_i + 1$
$d_{i+1} = d_i + 2(dy-dx)$ }

sinon { $x_{i+1} = x_i + 1$
$y_{i+1} = y_i$
$d_{i+1} = d_i + 2dy$ }

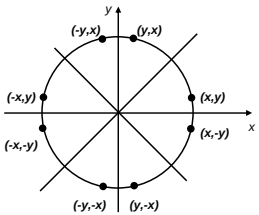
Algorithme de Bresenham (segment)

- Cet algorithme est valable uniquement pour les segments de 0 à 45 degrés.

- Une version similaire est définie pour les segments
  - de 45° à 90°
  - de 90° à 135°
  - de 135° à 180°.

3.3 Tracé de cercle

- **Tracé sur 45° puis utilisation des symétries**



- **équations du cercle de rayon r :**

$$y = \sqrt{r^2 - x^2}$$

$$\begin{cases} x = r\cos\theta \\ y = r\sin\theta \end{cases}$$

x **varie de** 0 **à** $r/\sqrt{2}$

θ **varie de** 0 **à** $\pi/4$

- calculs de racines et de carrés
- calculs de cosinus et sinus

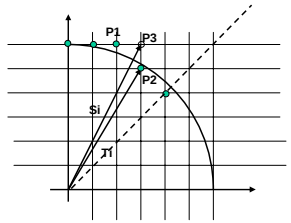
Algorithme de Bresenham (cercle)

- **opérations simples: multiplication par des puissances de 2 additions et soustractions**
- **calculs sur l'arc de cercle entre 90° et 45°, puis utilisation des symétries** ⇒ **directions** $+x$ **et** $-y$

$$Si = \sqrt{(x_{i-1}+1)^2 + (y_{i-1})^2}$$
$$Ti = \sqrt{(x_{i-1}+1)^2 + (y_{i-1}-1)^2}$$

$$D(Si) = Si^2 - r^2 \quad D(Si) \geq 0$$
$$D(Ti) = Ti^2 - r^2 \quad D(Ti) \leq 0$$

$$di = D(Si) + D(Ti)$$



Algorithme de Bresenham (cercle)

- **init :** $d_1 = 3 - 2r$
- **algo :**

si $d_i >= 0$ alors /* le cercle passe près de P2*/
{ $x_{i+1} = x_i + 1$
$y_{i+1} = y_i - 1$
$d_{i+1} = d_i + 4(x_i - y_i) + 10$ }

si $d_i < 0$ alors /* le cercle passe près de P3*/
{ $x_{i+1} = x_i + 1$
$y_{i+1} = y_i$
$d_{i+1} = d_i + 4x_i + 6$ }

3.4 Tracé d'ellipse

- **Fonction polynomiale du second degré**

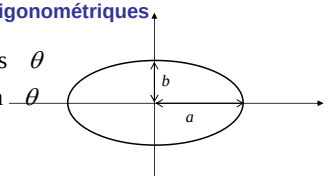
$$F(x, y) = b^2x^2 + a^2y^2 - a^2b^2 = 0$$

$$y = b\sqrt{1 - x^2/a^2}$$

où $2a$ représente la longueur du grand axe et $2b$ celle du petit axe

- **Fonctions trigonométriques**

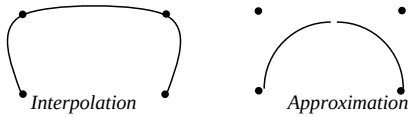
$$x = a\cos\theta$$
$$y = b\sin\theta$$



3.5 Tracé de courbes

- **Courbe donnée par** $y=f(x)$
  il suffit de calculer les valeurs de y pour $x \in D$

- **Courbe correspondant à la forme sous-tendue par les n+1 points:** $P_0, \ldots P_n$
  - la courbe passe par ces points : interpolation
  - la courbe se rapproche de ces points : approximation



- **Nous traiterons les courbes d 'approximation**
  - courbes de Bézier
  - courbes B-splines

Les courbes de Bézier

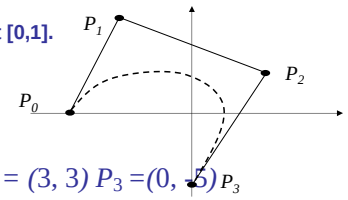
- **Soient n+1 points de contrôle** $P_0, \ldots P_n$ **définis par leurs coordonnées (x,y). On définit le polynôme de Bézier par l'équation vectorielle suivante:**

$$P(t) = \sum_{i=0}^{n} P_i \cdot B_{i,n}(t)$$

avec $B_{i,n}(t) = \dfrac{n!}{i!(n-i)!} t^i (1-t)^{n-i}$

- **L'intervalle de variation de t est [0,1].**

$P(0) = P_0$ *et* $P(1) = P_n$



- **Exemple**

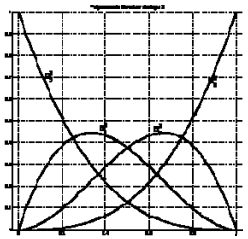
$P_0 = (-5,0)$, $P_1 = (-3, 7)$, $P_2 = (3, 3)$ $P_3 = (0, -6)$

Propriétés des courbes de Bézier

- **Le nombre de points de contrôle détermine l'ordre du polynôme qui définit la courbe.**

  degré de P = n-1

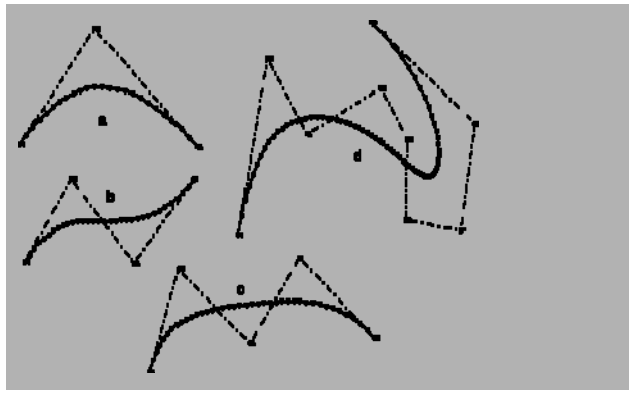
- **Chaque point effectue un contrôle global sur la courbe.**



Propriétes des courbes de Bézier

- **Une courbe de Bézier passe par** P_0 **et** P_n **(en général elle ne passe pas par les autres points).**
- **La courbe est tangente aux cotés du polygone en ses points extrêmes.**
- **Chaque point de contrôle exerce une attraction sur la portion de la courbe qui se trouve près de lui.**
- **La formulation de la courbe de Bézier autorise la création de courbes fermées et de courbes qui se recoupent.**
- **Il est garanti qu 'une courbe de Bézier sera toujours contenue à l 'intérieur du contour polygonal convexe donné par** $P_0, \ldots P_n$.
- **Il est possible de relier 2 courbes de Bézier.**
- **Il est possible de multiplier les points de contrôle.**

Exemples de courbes de Bézier



Les courbes B-splines

- **Formulation semblable à celle des courbes de Bézier. L'équation vectorielle d'une courbe spline d 'ordre j :**

$$P(t) = \sum_{i=0}^{n} P_i \cdot N_{i,j}(t)$$

$$N_{i,1}(t) = \begin{cases} 1 & \text{si} \quad x_i \le t < x_{i+1} \\ 0 & \text{ailleurs} \end{cases}$$

$$N_{i,j}(t) = \frac{(t - x_i) \cdot N_{i,j-1}(t)}{x_{i+j-1} - x_i} + \frac{(x_{i+j} - t) \cdot N_{i+1,j-1}(t)}{x_{i+j} - x_{i+1}}$$

puisque dans cette formule, les dénominateurs peuvent devenir nuls, on adopte la convention que 0/0 = 0 (cas de la multiciplité des points de contrôle)

Calcul d 'un vecteur nœud

- **l'ordre j de la courbe est toujours un entier :** $2 \le j \le n+1$

$$\begin{cases} x_i = 0 & \text{si} \quad i < j \\ x_i = i - j + 1 & \text{si} \quad j \le i \le n \\ x_i = n - j + 2 & \text{si} \quad i > n \end{cases}$$

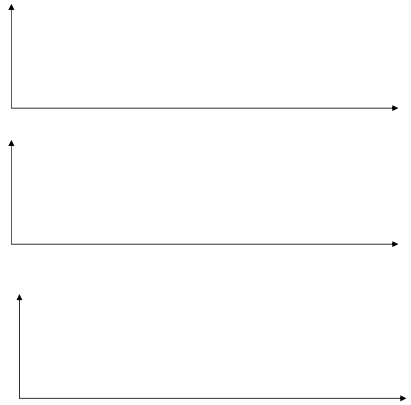
- **l'intervalle de variation de t est** $[0, t_{max}]$ $t_{max} = n - j + 2$

- **les valeurs de** x_i **sont les éléments d'un vecteur nœud**

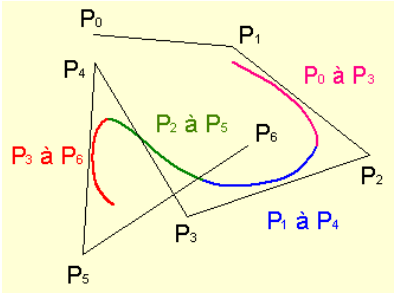
- **un vecteur nœud est une suite d'entiers positifs** x_i, $x_i \le x_{i+1}$

n	*j*	*vecteur noeud*	t_{max}
3	2	0 0 1 2 3 3	3
	3	0 0 0 1 2 2 2	2
	4	0 0 0 0 1 1 1 1	1
6	2	0 0 1 2 3 4 5 6 6	6
	3	0 0 0 1 2 3 4 5 5 5	5
	4	0 0 0 0 1 2 3 4 4 4 4	4
	5	0 0 0 0 0 1 2 3 3 3 3 3	3

Courbes obtenues pour n=3



Exemples



http://raphaello.univ-fcomte.fr/IG/Lissage/Exemples/BSpline.htm

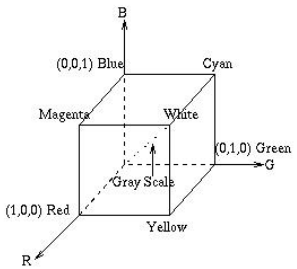
Exemple interactif de courbe de bezier et B-spline
http://www.mat.ulaval.ca/anum/ch4/html/interpolation/node4a.html

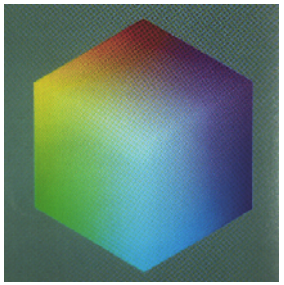
Propriétés des B-splines

- Le contrôle de la courbe est local: chaque sommet affecte la forme de la courbe dans un intervalle donné
- la base permet de changer l'ordre de la courbe sans changer le nombre de points.
- Si l'ordre de la courbe égale le nombre de points de contrôle, alors la courbe obtenue est identique à la courbe de Bézier correspondante.
- La technique des B-splines permet de créer des courbes avec des coins : il suffit d 'introduire des points multiples.

3.6 Couleur d'un pixel

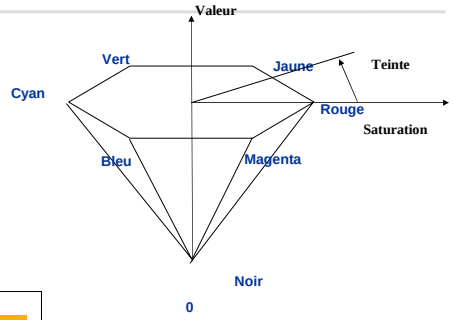
- Modèle RGB





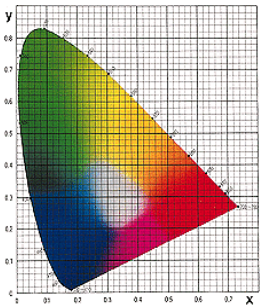
HSV

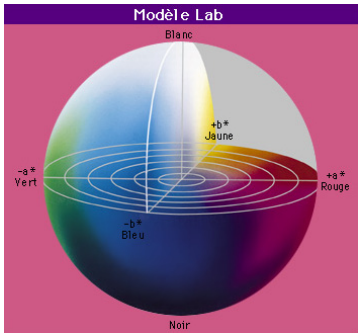
- Teinte (Hue)
- Saturation
- Valeur



Exemple : palette PowerPoint

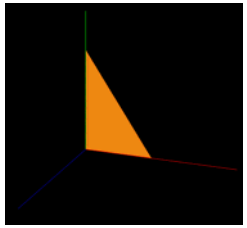
CIE





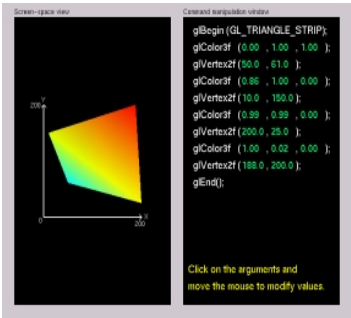
Exemple OpenGL

```
//Utilisation du mode RGB ou RGBA en
  OpenGL
glutInitDisplayMode(GLUT_RGB)
glColor3f(.9,.5,.1);
glBegin(GL_POLYGON);
    {
      glVertex3i(0,0,0);
      glVertex3i(2,0,0);
      glVertex3i(0,3,0);
     }
glEnd()
```



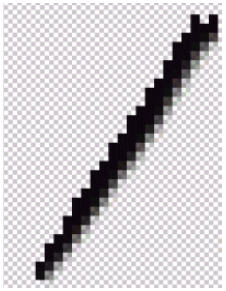
Exemple OpenGL

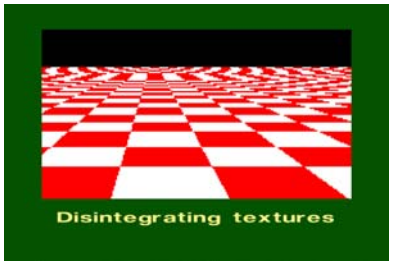
exemple issu du tutoriel de Nate Robins :
http://www.xmission.com/~nate/tutors.html



3.7 Antialiasing

- Méthodes de filtrage pour éviter l'effet créneaux et déformations de textures

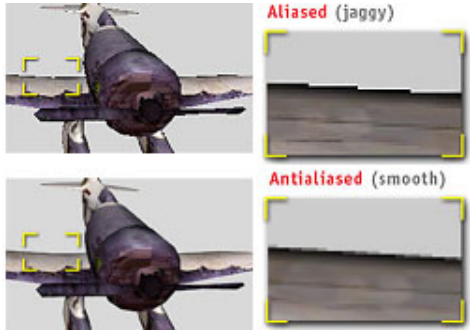




Antialiasing

. Traitements intégrés dans les cartes graphiques

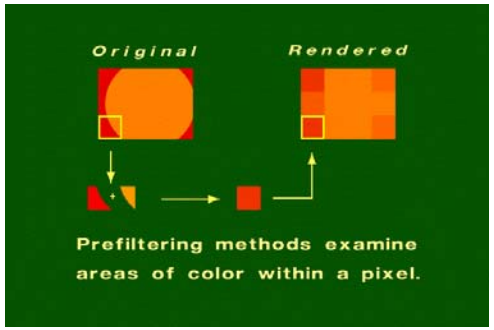
Ex : NVidia



Antialiasing

. préfiltrage

(siggraph93 Rosalee Nerheim-Wolfe)

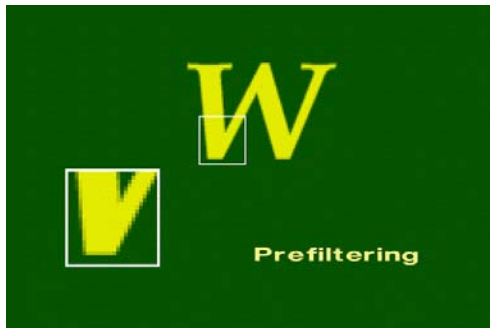


Antialiasing

. exemple

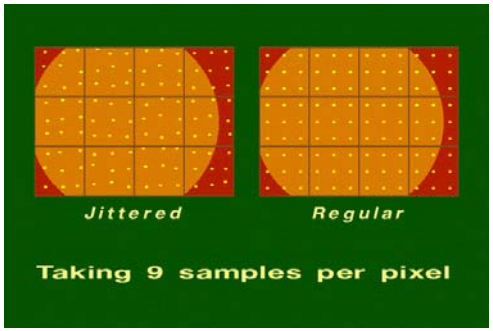
Antialiasing

. exemple



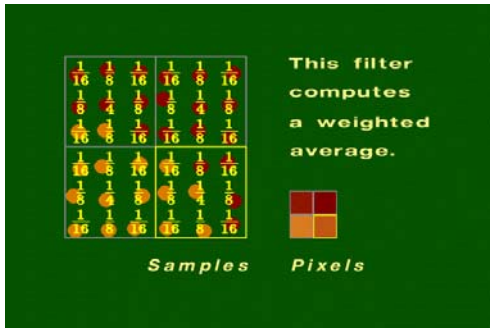
Antialiasing

. Postfiltrage (suréchantillonnage)



Antialiasing

. Postfiltrage (suréchantillonnage)



Antialiasing

. Exemple OpenGL : lissage au niveau des primitives + mélange des couleurs (blending)

- `glEnable(GL_POINT_SMOOTH);`
- `glEnable(GL_LINE_SMOOTH);`
- `glEnable(GL_POLYGON_SMOOTH);`