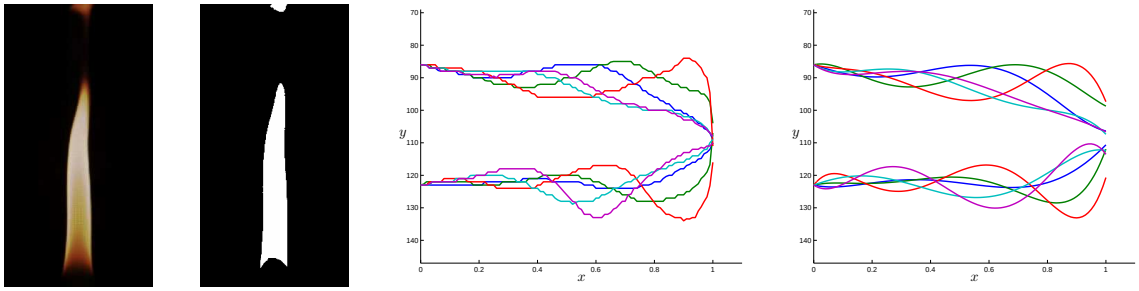
TP2 – Estimation de paramètres

Description des données

Les données utilisées dans ce TP sont une séquence réelle de $n = 10$ images d'une flamme de bougie, auxquelles sont appliquées une certain nombre de prétraitements : après seuillage, chaque image de flamme (cf. figure 1-a) fournit une image binaire (cf. figure 1-b) dans laquelle la silhouette de la flamme est détectée. Après normalisation, toutes les silhouettes ont une même hauteur égale à 1. Enfin, ces silhouettes sont tournées d'un quart de tour vers la droite (cf. figure 1-c).



(a) (b) (c) (d)

FIGURE 1 – (a) Une des $n = 10$ images de la séquence réelle. (b) Détection de la silhouette par seuillage. (c) Silhouettes de la séquence d'images, après normalisation et rotation d'un quart de tour vers la droite. (d) Modélisation des silhouettes par des paires de courbes de Bézier.

Lancez le script `donnees`, qui charge deux matrices `bord_inf` et `bord_sup` de taille $p \times n$, où $p = 101$ désigne le nombre de points de chaque bord de chaque flamme, de telle sorte que `bord_inf(i,j)` et `bord_sup(i,j)`, avec $(i, j) \in \{1, \ldots, p\} \times \{1, \ldots, n\}$, contiennent les ordonnées des bords inférieur et supérieur de la $j^{\text{ème}}$ silhouette, à l'abscisse $x_i = \frac{i-1}{p-1} \in [0, 1]$. La figure 1-c montre que ces silhouettes ont toutes la même base, c'est-à-dire que les ordonnées `bord_inf(1,j)` et `bord_sup(1,j)` ne dépendent pas de j et valent, respectivement, 123 et 86.

Description du modèle

Chaque bord de chaque silhouette peut être modélisé par une *courbe de Bézier* de degré d, entièrement définie par $d+1$ points de contrôle, dont les abscisses $\alpha_k = k/d$, $k \in \{0, \ldots, d\}$, sont équiréparties dans l'intervalle $[0, 1]$. Les points de contrôle sont notés $P_k^j = (\alpha_k, \beta_k^j)$ pour le bord inférieur de la $j^{\text{ème}}$ silhouette, et $Q_k^j = (\alpha_k, \gamma_k^j)$ pour son bord supérieur. Les ordonnées $\beta_0^j = 123$ et $\gamma_0^j = 86$ étant indépendantes de j, sont notées β_0 et γ_0. Les bords de la silhouette numéro j sont donc modélisés par les courbes de Bézier suivantes :

$$\left\{ \begin{array}{l} y_i^j = f_{\text{inf}}^j(x_i) = \beta_0 \, B_0^d(x_i) + \displaystyle\sum_{k=1}^{d} \beta_k^j \, B_k^d(x_i) \\ y_i^j = f_{\text{sup}}^j(x_i) = \gamma_0 \, B_0^d(x_i) + \displaystyle\sum_{k=1}^{d} \gamma_k^j \, B_k^d(x_i) \end{array} \right\}, \quad \forall\, i \in \{1, \ldots, p\} \tag{1}$$

Dans ces expressions, les fonctions B_k^d, $k \in \{0, \ldots, d\}$, appelées *polynômes de Bernstein*, sont définies par :

$$B_k^d(x_i) = \binom{d}{k} \, x_i^k (1 - x_i)^{d-k} \tag{2}$$

où $\binom{d}{k}$, nombre de combinaisons de k objets parmi d, se calcule avec la fonction `nchoosek` de Matlab.

Exercice 1 : estimation des paramètres de courbes de Bézier

Modéliser la silhouette numéro j consiste à estimer les d paramètres $\beta^j = [\beta_1^j, \ldots, \beta_d^j]$ du bord inférieur et les d paramètres $\gamma^j = [\gamma_1^j, \ldots, \gamma_d^j]$ du bord supérieur, les ordonnées de ces bords étant stockées dans les matrices `bord_inf` et `bord_sup`. Les paramètres à estimer sont donc le degré d, commun à l'ensemble des courbes de Bézier, et les vecteurs β^j et γ^j. Ce problème d'estimation étant linéaire en β^j et γ^j, mais non linéaire en d, il semble raisonnable de faire une hypothèse sur la valeur du degré d et d'estimer seulement β^j et γ^j.

Établissez (sur papier) les expressions des matrices $\mathbf{A}_{\text{inf}}^j$ et $\mathbf{A}_{\text{sup}}^j$ et des vecteurs $\mathbf{B}_{\text{inf}}^j$ et $\mathbf{B}_{\text{sup}}^j$ caractérisant les systèmes linéaires suivants :

$$\begin{cases} \mathbf{A}_{\text{inf}}^j [\beta_1^j, \ldots, \beta_d^j]^\top = \mathbf{B}_{\text{inf}}^j \\ \mathbf{A}_{\text{sup}}^j [\gamma_1^j, \ldots, \gamma_d^j]^\top = \mathbf{B}_{\text{sup}}^j \end{cases} \tag{3}$$

qui traduisent le fait que chaque courbe de Bézier recherchée doit passer par p points d'abscisses x_i, $i \in \{1, \ldots, p\}$, dont les ordonnées $f_{\text{inf}}^j(x_i)$ pour le bord inférieur, $f_{\text{sup}}^j(x_i)$ pour le bord supérieur, sont données en (1). Notez bien que le nombre d'inconnues de ces équations est égal à d, et non pas à $d+1$, car β_0 et γ_0 sont supposés connus.

Écrivez la fonction `moindres_carres`, appelée par le script `exercice_1`, censée retourner la solution approchée des équations (3), au sens des moindres carrés, en utilisant l'une des méthodes présentées en cours (pseudo-inverse, SVD, factorisation QR) ou l'opérateur \ (lisez la documentation).

Écrivez ensuite la fonction `bezier` permettant de calculer les expressions (1) des ordonnées $f_{\text{inf}}^j(x_i)$ et $f_{\text{sup}}^j(x_i)$. Lancez le script `exercice_1` en testant différentes valeurs de d comprises entre 2 et 20. Dans la suite du TP, vous choisirez pour d la valeur 5, qui constitue un bon compromis entre ces valeurs extrêmes.

Exercice 2 : estimation de paires de courbes de Bézier

La figure 1-d montre que les modélisations correspondant aux silhouettes de la figure 1-c ne sont pas fermées au niveau du sommet. Pour que les deux courbes de Bézier se rejoignent au sommet de la flamme, il faut les coupler, c'est-à-dire faire en sorte que le point de contrôle situé à l'abscisse $x_p = 1$ soit commun aux deux courbes. Il faut donc reformuler le problème (3) sous la forme d'un système linéaire unique à $2d-1$ paramètres :

$$\mathbf{A}^j [\beta_1^j, \ldots, \beta_{d-1}^j, \gamma_1^j, \ldots, \gamma_{d-1}^j, \epsilon^j]^\top = \mathbf{B}^j \tag{4}$$

où le paramètre ϵ^j est égal à l'ordonnée du sommet de la flamme, c'est-à-dire que $\epsilon^j = \beta_d^j = \gamma_d^j$.

Après avoir établi (sur papier) les expressions des matrices $\mathbf{A}^j$ et des vecteurs $\mathbf{B}^j$ définissant le système (4), écrivez la fonction `moindres_carres_paire`, appelée par le script `exercice_2`, censée retourner sa solution approchée, au sens des moindres carrés.

Exercice 3 : application à la réalité virtuelle

L'analyse précédente permet de caractériser la silhouette d'une flamme de bougie par $2d-1$ paramètres. Pour simuler de nouvelles silhouettes, on peut modéliser l'ordonnée de chaque point de contrôle par une loi normale, puis utiliser les lois estimées pour tirer aléatoirement de nouveaux points de contrôle.

Écrivez la fonction `estimation_lois`, appelée par le script `exercice_3`, qui permet d'estimer la moyenne et l'écart-type de l'ordonnée de chaque point de contrôle, considérée comme une variable aléatoire.

Écrivez la fonction `tirage_aleatoire`, également appelée par le script `exercice_3`, qui tire des valeurs aléatoires pour les paramètres $[\beta_1, \ldots, \beta_{d-1}, \gamma_1, \ldots, \gamma_{d-1}, \epsilon]$, à l'aide des paramètres de lois normales précédemment estimés et de la fonction `randn`.

Lancez enfin le script `simulation_flamme`, qui constitue un exemple d'application de l'estimation de paramètres à la synthèse d'images, également appelée *réalité virtuelle*.