Chapitre V

Algorithme de calcul d'un arrangement de lignes polygonales équivalent à un arrangement de polycourbes

V.1. Introduction

Le résultat qui conclut le chapitre précédent justifie la recherche d'un algorithme de construction d'un arrangement de polygones de contrôle équivalent à un arrangement de polycourbes (de Bézier, mais comme à partir de maintenant nous ne parlerons que de polycourbes de Bézier, nous ne le préciserons plus). Mais le théorème ne nous fournit pas de critère utilisable dans un algorithme, tout comme dans le cas de l'enveloppe convexe, car les valeurs définies dans la preuve sont calculées en utilisant les courbes de Bézier elles-mêmes et non pas uniquement leurs polygones de contrôle. Il nous faudra ainsi trouver des conditions "polygonales" assurant l'équivalence des deux arrangements avant de présenter un algorithme.

Précisons encore une fois que nous ne calculons pas la structure combinatoire de l'arrangement de polycourbes, mais seulement un arrangement de lignes polygonales qui a cette structure combinatoire. Ainsi, notre algorithme pourrait être vu comme effectuant un prétraitement de l'arrangement de polycourbes, en vue des opérations ultérieures. Mais il est plus que cela, car, comme nous verrons à la fin de la section V.3, nous construisons en fait *deux* arrangements de lignes polygonale équivalents à celui de polycourbes, l'arrangement de polygones de contrôle, $\mathcal{A}(\mathcal{P})$, et l'arrangement de polygones support, $\mathcal{A}(\mathcal{S})$. Ces deux arrangements réalisent un "encadrement" pour celui de polycourbes et certains des problèmes de base sur des arrangements pourront être résolus en travaillant uniquement sur des lignes polygonales. Ainsi, si $c_{\mathcal{B}}$ est une cellule de $\mathcal{A}(\mathcal{B})$, alors $c_{\mathcal{P}} \cap c_{\mathcal{S}} \subset c_{\mathcal{B}} \subset c_{\mathcal{P}} \cup c_{\mathcal{S}}$, où $c_{\mathcal{P}}$ et $c_{\mathcal{S}}$ sont les cellules de $\mathcal{A}(\mathcal{P})$, respectivement $\mathcal{A}(\mathcal{S})$, qui correspondent à $c_{\mathcal{B}}$. En utilisant cette propriété, la localisation d'un point dans $\mathcal{A}(\mathcal{B})$ peut se faire sur $\mathcal{A}(\mathcal{P})$ et $\mathcal{A}(\mathcal{S})$, en subdivisant éventuellement certaines courbes de Bézier ; si la position n'est pas "trop" dégénérée, la solution est trouvée assez vite.

La section V.3 est consacrée à la recherche de critères d'équivalence de l'arrangement de polygones de contrôle à celui de polycourbes, et nous voyons tout de suite que des conditions posées uniquement sur les polygones de contrôle ne peuvent pas suffire. Nous avons ainsi besoin des polygones support, et en les utilisant nous donnons d'abord un résultat décrivant les propriétés que les polygones de contrôle et support doivent satisfaire pour l'équivalence, et ensuite nous prouvons que par subdivision de de Casteljau nous pouvons arriver à des écritures des polycourbes de l'arrangement donné dont les lignes polygonales associées satisfont les propriétés mentionnées. La section V.4 présente la structure de l'algorithme que nous avons conçu, en expliquant en détail son fonctionnement. Une étude de coût théorique y est aussi menée. La section V.5 contient les résultats obtenus en traitant un échantillon d'arrangements par l'implémentation de notre algorithme, ainsi que leurs interprétations. La section V.6 présente quelques exemples d'arrangements de polycourbes, avec les arrangements de polygones de contrôle et de polygones support qui leur sont associés.

V.2. Concentré de chapitre (V et fin)

Ce chapitre étant le pendant, pour le problème des arrangements, du chapitre trois, sa lecture se fait à peu près de la même façon.

Premier niveau. Dans la section V.3, ne lisez pas les preuves des résultats, mais les commentaires divers se trouvant entre les énoncés oui (et les énoncés bien évidemment).

Dans la section V.4, la lecture des sous sections V.4.1 et V.4.2 (sans leurs paragraphes) vous permet de vous faire une idée sur l'algorithme, après quoi la sous section V.4.3 fournit son coût. Les sections V.5 et V.6 sont à parcourir en totalité.
Deuxième niveau. À part les preuves des résultats de la section V.3 (surtout celle du lemme V.1, bien technique), tout est à lire.

V.3. Résultats théoriques

De même que pour le problème du calcul de l'enveloppe convexe d'une polycourbe de Bézier, le résultat donné par le théorème IV.24 ne peut pas être traduit par un algorithme respectant l'approche que nous adoptons, car le calcul des valeurs ε_0, ε_1, ε_2' et ε_2'' fait intervenir des courbes de Bézier et non pas uniquement leurs B-polygones. Nous devons donc trouver des critères d'équivalence entre un arrangement de polycourbes de Bézier et celui des polygones de contrôle respectifs reposant seulement sur ces derniers.

V.3.1. *Le nombre d'intersections*

Si $\mathcal{B}_1$ et $\mathcal{B}_2$ sont deux courbes de Bézier complétement convexes, aucune information sur le cardinal de leur intersection ne peut être obtenue en regardant l'intersection de leurs B-polygones. La figure V.1 illustre cette affirmation.

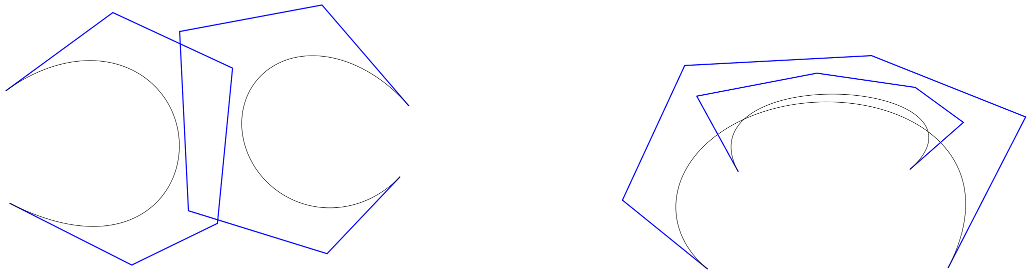


Figure V.1

Une courbe de Bézier étant "encadrée" d'un côté par son B-polygone et de l'autre côté par le segment joignant ses extrémités, il est naturel de chercher des informations supplémentaires sur l'intersection des deux courbes dans les intersections des segments $[\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}]$ et $[\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]$ entre eux et respectivement avec les lignes polygonales $\mathcal{P}_2$ et $\mathcal{P}_1$. Comme $\mathrm{card}([\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}] \cap [\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]) \in \{0,1\}$, nous cherchons en fait des critères assurant que les deux courbes sont disjointes ou que le cardinal de leur intersection est égal à 1.

Pour avoir $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) = 0$, il suffit de vérifier si $\mathcal{P}_1 \cap \mathcal{P}_2 = \emptyset$ et $reg(\mathcal{P}_1, [\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}]) \cap reg(\mathcal{P}_2, [\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]) = \emptyset$, car les propriétés des courbes de Bézier impliqueront alors $\mathcal{B}_1 \cap \mathcal{B}_2 = \emptyset$.

Si nous voulons nous assurer de la relation $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) = 1$, la condition la plus forte que nous pouvons poser sur les intersections des B-polygones et de segments $[\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}]$ et $[\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]$ est

$$\begin{aligned}\mathrm{card}(\mathcal{P}_1 \cap \mathcal{P}_2) &= \mathrm{card}([\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}] \cap [\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]) \\ &= \mathrm{card}([\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}] \cap \mathcal{P}_2) = \mathrm{card}(\mathcal{P}_1 \cap [\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]) = 1\end{aligned} \tag{V.1}$$

et nous pouvons voir figure V.2 qu'elle n'est pas suffisante.

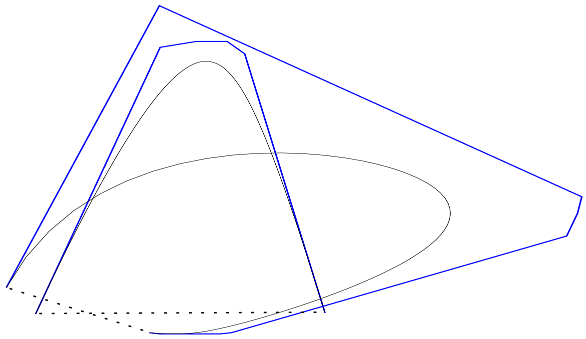


Figure V.2

Le résultat suivant nous donne une condition suffisante pour que l'intersection des deux courbes de Bézier ait le cardinal égal à 1 :

Lemme V.1. – *Soient $\mathcal{B}_1$ et $\mathcal{B}_2$ deux courbes de Bézier complétement convexes telles que $\mathcal{A}(\{\mathcal{B}_1, \mathcal{B}_2\})$ soit un arrangement simple. Supposons que leurs B-polygones respectifs, $\mathcal{P}_1$ et $\mathcal{P}_2$, satisfont la relation (V.1). Soient $i_1 \in \{0, \ldots, m_1 - 1\}$, $i_2 \in \{0, \ldots, m_2 - 1\}$ tels que $\mathcal{P}_1 \cap \mathcal{P}_2 = [\mathrm{P}^{(1)}_{i_1}\mathrm{P}^{(1)}_{i_1+1}] \cap [\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}]$. Les conditions suivantes sont suffisantes pour avoir* $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) = 1$ *:*

$$\left(\begin{array}{c} \left(\begin{array}{c} \mathrm{P}^{(1)}_{i_1} \in \mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2) \\ ou \\ \mathrm{P}^{(1)}_{i_1+1} \in \mathrm{DP}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2) \end{array}\right) \Longrightarrow \mathrm{P}^{(1)}_0 \in \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2)} \\ et \\ \left(\begin{array}{c} \mathrm{P}^{(1)}_{i_1} \in \mathrm{DP}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2) \\ ou \\ \mathrm{P}^{(1)}_{i_1+1} \in \mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2) \end{array}\right) \Longrightarrow \mathrm{P}^{(1)}_{m_1} \in \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2)} \end{array}\right) \tag{V.2}$$

et

$$\left(\begin{array}{c} \left(\begin{array}{c} \mathrm{P}^{(2)}_{i_2} \in \mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(1)}_{i_1}\mathrm{P}^{(1)}_{i_1+1}, \mathcal{P}_1) \\ ou \\ \mathrm{P}^{(2)}_{i_2+1} \in \mathrm{DP}(\mathrm{P}^{(1)}_{i_1}\mathrm{P}^{(1)}_{i_1+1}, \mathcal{P}_1) \end{array}\right) \Longrightarrow \mathrm{P}^{(2)}_0 \in \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(1)}_{i_1}\mathrm{P}^{(1)}_{i_1+1}, \mathcal{P}_1)} \\ et \\ \left(\begin{array}{c} \mathrm{P}^{(2)}_{i_2} \in \mathrm{DP}(\mathrm{P}^{(1)}_{i_1}\mathrm{P}^{(1)}_{i_1+1}, \mathcal{P}_1) \\ ou \\ \mathrm{P}^{(2)}_{i_2+1} \in \mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(1)}_{i_1}\mathrm{P}^{(1)}_{i_1+1}, \mathcal{P}_1) \end{array}\right) \Longrightarrow \mathrm{P}^{(2)}_{m_2} \in \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(1)}_{i_1}\mathrm{P}^{(1)}_{i_1+1}, \mathcal{P}_1)} \end{array}\right) \tag{V.3}$$

La condition (V.2) est l'expression formelle de la propriété qui intuitivement est formulée comme suit : si nous prenons sur $\mathcal{P}_1$ un sens de parcours tel que cette ligne polygonale “sorte” de $\mathcal{P}_2$ quand ce sens est respecté, alors l'extrémité se trouvant “à la fin” de $\mathcal{P}_1$ par rapport au sens de parcours est située dans le demi-plan fermé délimité par $\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}$ qui ne contient pas $\mathcal{P}_2$ (voir figure V.3). La condition (V.3) est obtenue de (V.2) en échangeant les rôles de $\mathcal{P}_1$ et $\mathcal{P}_2$.

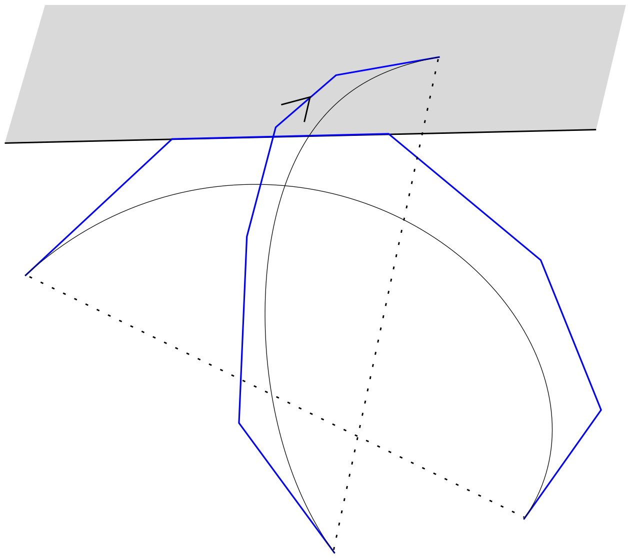


Figure V.3

Preuve du lemme – Soit $\{Q\} = \mathcal{P}_1 \cap \mathcal{P}_2$. Nous traitons séparément les trois cas suivants :

Cas 1. $Q \in \{P_0^{(1)}, P_{m_1}^{(1)}\} \cap \{P_0^{(2)}, P_{m_2}^{(2)}\}$

Cas 2. $Q \in \{P_0^{(1)}, P_{m_1}^{(1)}\} \cap \big(\mathcal{P}_2 \setminus \{P_0^{(2)}, P_{m_2}^{(2)}\}\big)$ ou $Q \in \{P_0^{(2)}, P_{m_2}^{(2)}\} \cap \big(\mathcal{P}_1 \setminus \{P_0^{(1)}, P_{m_1}^{(1)}\}\big)$

Cas 3. $Q \notin \{P_0^{(1)}, P_{m_1}^{(1)}\} \cup \{P_0^{(2)}, P_{m_2}^{(2)}\}$

Cas 1. Supposons que $Q = P_0^{(1)} = P_0^{(2)}$; les trois autres situations possibles se résolvent de la même manière.

Dans ce cas, nous avons

$$reg(\mathcal{P}_1, [P_0^{(1)} P_{m_1}^{(1)}]) \cap reg(\mathcal{P}_2, [P_0^{(2)} P_{m_2}^{(2)}]) = \emptyset$$

car dans le cas contraire nous obtiendrons

$$\mathrm{card}\big((\mathcal{P}_1 \cup [P_0^{(1)} P_{m_1}^{(1)}]) \cap (\mathcal{P}_2 \cup [P_0^{(2)} P_{m_2}^{(2)}])\big) \geq 2$$

Comme $P_0^{(1)} \in \mathcal{P}_1 \cap \mathcal{P}_2$, $P_0^{(1)} \in \mathcal{P}_1 \cap [P_0^{(2)} P_{m_2}^{(2)}]$, $P_0^{(1)} \in [P_0^{(1)} P_{m_1}^{(1)}] \cap \mathcal{P}_2$ et $P_0^{(1)} \in [P_0^{(1)} P_{m_1}^{(1)}] \cap [P_0^{(2)} P_{m_2}^{(2)}]$, ceci mène à une contradiction avec la condition (V.1).

Nous déduisons donc $\mathcal{B}_1 \cap \mathcal{B}_2 \subset \{P_0^{(1)}, P_{m_1}^{(1)}\} \cap \{P_0^{(2)}, P_{m_2}^{(2)}\}$ et par conséquent $\mathcal{B}_1 \cap \mathcal{B}_2 = \{P_0^{(1)}\}$.

Cas 2. Les deux situations de ce cas sont symétriques, il suffit donc de discuter la première.

Sans perte de généralité, supposons que $Q = P_0^{(1)} \in \mathcal{P}_2 \setminus \{P_0^{(2)}, P_{m_2}^{(2)}\}$. Ce cas est présenté dans la figure V.4.

Comme $P_0^{(1)} \in [P_0^{(1)} P_{m_1}^{(1)}] \cap \mathcal{P}_2$, nous déduisons que le point d'intersection des segments $[P_0^{(1)} P_{m_1}^{(1)}]$ et $[P_0^{(2)} P_{m_2}^{(2)}]$ est un point de $]P_0^{(2)} P_{m_2}^{(2)}[$.

Il est évident que $[P_0^{(1)} P_{m_1}^{(1)}] \cap \big(\mathcal{B}_2 \setminus \{P_0^{(2)}, P_{m_2}^{(2)}\}\big) \neq \emptyset$, car $[P_0^{(1)} P_{m_1}^{(1)}] \cap \mathcal{P}_2 \neq \emptyset$ et $[P_0^{(1)} P_{m_1}^{(1)}] \cap]P_0^{(2)} P_{m_2}^{(2)}[\neq \emptyset$. Nous voulons prouver que $\mathcal{P}_1 \cap \mathcal{B}_2 \neq \emptyset$.

Si $P_1^{(1)} \in DP(P_{i_2}^{(2)} P_{i_2+1}^{(2)}, \mathcal{P}_2)$, il résulte que $\mathcal{P}_1$ partage $reg(\mathcal{P}_2, [P_0^{(2)} P_{m_2}^{(2)}])$ en deux régions connexes disjointes, une connectée à $P_0^{(2)}$ et une connectée à $P_{m_2}^{(2)}$, car $\mathcal{P}_1 \cap]P_0^{(2)} P_{m_2}^{(2)}[\neq \emptyset$. Comme $\mathcal{B}_2$ est une courbe (connexe) d'extrémités $P_0^{(2)}$ et $P_{m_2}^{(2)}$ et qui est contenue dans $reg(\mathcal{P}_2, [P_0^{(2)} P_{m_2}^{(2)}])$, il vient $\mathcal{P}_1 \cap \mathcal{B}_2 \neq \emptyset$.

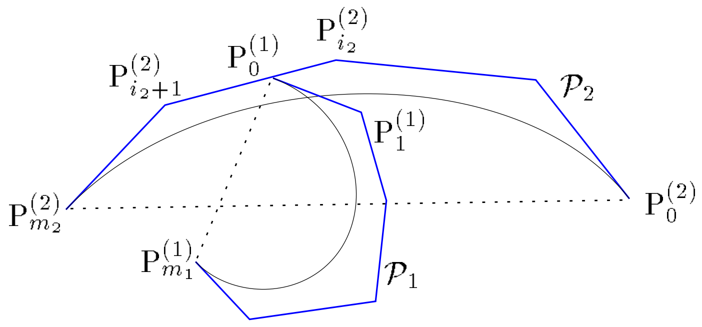


Figure V.4

Si $\mathrm{P}^{(1)}_1 \in \mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2)$, (V.2) implique $\mathrm{P}^{(1)}_{m_1} \in \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2)}$. D'un autre côté, les relations $[\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_{m_1}]\cap]\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_{m_2}[\neq \emptyset$ et $\mathrm{P}^{(1)}_0 \in \mathrm{DP}(\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_{m_2}, \mathcal{P}_2)$ donnent $\mathrm{P}^{(1)}_{m_1} \in \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_{m_2}, \mathcal{P}_2)}$; il vient

$$\mathrm{P}^{(1)}_{m_1} \in \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_{m_2}, \mathcal{P}_2)} \cap \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2)}$$

Mais $\mathcal{P}_2$ est convexe, par conséquent

$$\begin{aligned}\overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_{m_2}, \mathcal{P}_2)} \cap \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2)} \subset \\ \left(\overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_{m_2}, \mathcal{P}_2)} \cap \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_1, \mathcal{P}_2)}\right) \cup \\ \left(\overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_{m_2}, \mathcal{P}_2)} \cap \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{m_2}\mathrm{P}^{(2)}_{m_2-1}, \mathcal{P}_2)}\right)\end{aligned}$$

Quel que soit, parmi les deux ensembles dont l'union forme le terme droit de l'inclusion ci-dessus, l'ensemble auquel $\mathrm{P}^{(1)}_{m_1}$ appartient, il est évident, de par la convexité de $\mathcal{P}_2$ et le fait que $\mathrm{P}^{(1)}_0 \in \mathcal{P}_2 \setminus \{\mathrm{P}^{(2)}_0, \mathrm{P}^{(2)}_{m_2}\}$, que dans cette situation $[\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_{m_1}]\cap]\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_{m_2}[= \emptyset$, d'où une contradiction. Le point $\mathrm{P}^{(1)}_1$ ne peut ainsi pas appartenir au demi-plan $\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(2)}_{i_2}\mathrm{P}^{(2)}_{i_2+1}, \mathcal{P}_2)$ si (V.1) et (V.2) sont satisfaites et par conséquent $\mathcal{B}_2 \cap \mathcal{P}_1 \neq \emptyset$.

Ainsi, nous avons $[\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_{m_1}] \cap \left(\mathcal{B}_2 \setminus \{\mathrm{P}^{(2)}_0, \mathrm{P}^{(2)}_{m_2}\}\right) \neq \emptyset$ et $\mathrm{P}_1 \cap \mathcal{B}_2 \neq \emptyset$. Les points $\mathrm{P}^{(2)}_0$ et $\mathrm{P}^{(2)}_{m_2}$ ne peuvent pas appartenir à $reg(\mathcal{P}_1, [\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_{m_1}])$, car dans ce cas $[\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_{m_1}]\cap]\mathrm{P}^{(2)}_0\mathrm{P}^{(2)}_{m_2}[= \emptyset$, donc $\mathcal{B}_2$ traverse la région $reg(\mathcal{P}_1, [\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_{m_1}])$ en la partageant en deux parties connexes disjointes, l'une connectée à $\mathrm{P}^{(1)}_0$ et l'autre connectée à $\mathrm{P}^{(1)}_{m_1}$; il en résulte $\mathcal{B}_1 \cap \mathcal{B}_2 \neq \emptyset$, car $\mathcal{B}_1$ est une courbe (continue) d'extrémités $\mathrm{P}^{(1)}_0$ et $\mathrm{P}^{(1)}_{m_1}$ incluse dans $reg(\mathcal{P}_1, [\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_{m_1}])$.

Pour finir la discussion du cas 2, il nous reste à montrer $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) < 2$; la preuve sera faite par l'absurde, nous supposons que $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) \geq 2$ et nous montrons comment ceci nous mène à une contradiction.

Supposons que $\mathrm{P}^{(2)}_{i_2} \in \mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_1, \mathcal{P}_1)$ ou $\mathrm{P}^{(2)}_{i_2+1} \in \mathrm{DP}(\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_1, \mathcal{P}_1)$; nous obtiendrons, de par la condition (V.3), $\mathrm{P}^{(2)}_0 \in \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_1, \mathcal{P}_1)}$. Si $\mathrm{P}^{(2)}_{i_2} \in \mathrm{DP}(\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_1, \mathcal{P}_1)$ ou $\mathrm{P}^{(2)}_{i_2+1} \in \mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_1, \mathcal{P}_1)$, la démonstration est similaire.

Le point $\mathrm{P}^{(2)}_0$ appartient donc à $\overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_1, \mathcal{P}_1)}$. Nous pouvons en déduire que si Δ est une droite qui coupe $\mathcal{P}_2\big|_{[\mathrm{P}^{(2)}_0, \mathrm{P}^{(1)}_0]}$ en au moins deux points, alors $\Delta \cap \mathcal{P}_1 \subset [\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_1]$, car $\mathcal{P}_2\big|_{[\mathrm{P}^{(2)}_0, \mathrm{P}^{(1)}_0]}$ est convexe. D'un autre côté, si une droite Δ coupe $\mathcal{P}_2\big|_{[\mathrm{P}^{(1)}_0, \mathrm{P}^{(2)}_{m_2}]}$ deux fois (ou selon une de ses arêtes), il est évident que $\Delta \cap \mathcal{P}_1 \subset \{\mathrm{P}^{(1)}_0\}$, car $\mathrm{P}^{(2)}_{m_2} \in \mathrm{DP}_{\mathrm{op}}(\mathrm{P}^{(1)}_0\mathrm{P}^{(1)}_{m_1}, \mathcal{P}_1)$

et $\mathrm{P}_{m_1}^{(1)} \notin \mathrm{P}_0^{(1)}\mathrm{P}_{m_2}^{(2)}$.

Si $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) \geq 2$, alors il existe une droite Δ_0 (engendrée par exemple par deux points d'intersection de $\mathcal{B}_1$ et $\mathcal{B}_2$ distincts) telle que $\mathrm{card}(\Delta_0 \cap \mathcal{B}_1) = 2$ et $\mathrm{card}(\Delta_0 \cap \mathcal{B}_2) = 2$ (le cardinal de l'intersection d'une droite avec une courbe de Bézier complètement convexe ne peut pas être supérieur à 2). La propriété de la diminution de la variation et la convexité des polygones $\mathcal{P}_1$ et $\mathcal{P}_2$ impliquent alors $\mathrm{card}(\Delta_0 \cap \mathcal{P}_1) = 2$ et $\mathrm{card}(\Delta_0 \cap \mathcal{P}_2) = 2$.

Nous avons vu que la droite Δ_0 ne peut pas couper $\mathcal{P}_2\big|_{[\mathrm{P}_0^{(2)},\mathrm{P}_0^{(1)}]}$ en deux points, ou couper $\mathcal{P}_2\big|_{[\mathrm{P}_0^{(1)},\mathrm{P}_{m_2}^{(2)}]}$ en deux points, car alors le cardinal de son intersection avec $\mathcal{P}_1$ ne serait pas 2. Δ_0 doit par conséquent couper $\mathcal{P}_2\big|_{[\mathrm{P}_0^{(2)},\mathrm{P}_0^{(1)}[}$ en un point, soit Q_1 ce point, et $\mathcal{P}_2\big|_{]\mathrm{P}_0^{(1)},\mathrm{P}_{m_2}^{(2)}]}$ en un point, notons-le Q_2. Comme $\mathcal{P}_2$ est convexe, nous aurons $]\mathrm{Q}_1\mathrm{Q}_2[\subset reg(\mathcal{P}_2, [\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}])$. Mais nous avons vu que le segment $[\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}]$ divise $reg(\mathcal{P}_2, [\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}])$ en deux parties connexes disjointes, l'une étant connectée à $\mathcal{P}_2\big|_{[\mathrm{P}_0^{(2)},\mathrm{P}_0^{(1)}[}$ et l'autre étant connectée à $\mathcal{P}_2\big|_{]\mathrm{P}_0^{(1)},\mathrm{P}_{m_2}^{(2)}]}$. Nous aurons donc $]\mathrm{Q}_1\mathrm{Q}_2[\cap]\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}[\neq \emptyset$ et il vient $\mathrm{Q}_1\mathrm{Q}_2\cap]\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}[\neq \emptyset$. Évidemment, la droite $\mathrm{Q}_1\mathrm{Q}_2$ ne peut pas être égale à la droite $\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}$, cette dernière coupant $\mathcal{P}_2$ en un seul point. Il résulte $\mathrm{card}(\Delta_0 \cap \mathcal{P}_1) \leq 1$, car $\mathcal{P}_1$ est un polygone convexe et par conséquent toute droite coupe $\mathcal{P}_1\cup]\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}[$ en au plus deux points, et nous avons ainsi obtenu une contradiction avec la définition de Δ_0. Nous pouvons donc conclure que $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) = 1$ et la preuve pour le cas 2 est finie.

Cas 3. Cette situation, la plus générale, est présentée dans la figure V.5.

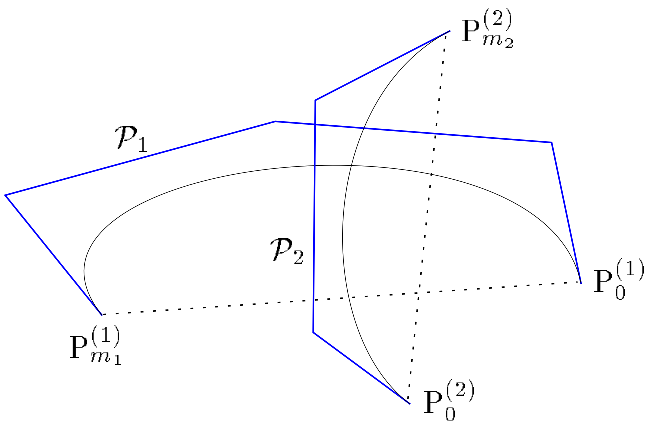


Figure V.5

Nous pouvons avoir $[\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}] \cap [\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}] \in \{\mathrm{P}_0^{(1)}, \mathrm{P}_{m_1}^{(1)}\}$ ou $[\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}] \cap [\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}] \in \{\mathrm{P}_0^{(2)}, \mathrm{P}_{m_2}^{(2)}\}$, mais bien sûr pas les deux simultanément (nous serions dans le cas 1). Supposons que $[\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}] \cap [\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}] \notin \{\mathrm{P}_0^{(2)}, \mathrm{P}_{m_2}^{(2)}\}$ et pour fixer les notations admettons que $\mathrm{P}_0^{(2)} \in \mathrm{DP}_{\mathrm{op}}(\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}, \mathcal{P}_1)$ et $\mathrm{P}_{m_2}^{(2)} \in \mathrm{DP}(\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}, \mathcal{P}_1)$ et que $\mathrm{P}_{m_1}^{(1)} \in \mathrm{DP}(\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}, \mathcal{P}_2)$ et $\mathrm{P}_0^{(1)} \in \overline{\mathrm{DP}_{\mathrm{op}}(\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}, \mathcal{P}_2)}$.

Comme pour le cas 2, il est facile à voir que $[\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}] \cap \mathcal{B}_2 \neq \emptyset$ et à montrer que $\mathcal{P}_1 \cap \mathcal{B}_2 \neq \emptyset$. Nous avons alors de nouveau la courbe $\mathcal{B}_2$ qui traverse $reg(\mathcal{P}_1, [\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}])$ et nous en déduisons $\mathcal{B}_1 \cap \mathcal{B}_2 \neq \emptyset$. Comme pour le cas précédent, nous démontrons que les deux courbes se coupent en exactement un point par l'absurde, en supposant $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) \geq 2$

et en obtenant une contradiction.

Soient $Q_1 \neq Q_2 \in \mathcal{B}_1 \cap \mathcal{B}_2$. Si $reg(\mathcal{B}_1|_{[Q_1,Q_2]}, \mathcal{B}_2|_{[Q_1,Q_2]})$ était convexe, nous pourrions en déduire de la convexité des courbes $\mathcal{B}_1$ et $\mathcal{B}_2$ que

$$\big(\mathcal{B}_1 \cup \{P_0^{(1)}, P_{m_1}^{(1)}\}\big) \cap \big(\mathcal{B}_2 \cup \{P_0^{(2)}, P_{m_2}^{(2)}\}\big) = \{Q_1, Q_2\}$$

ce qui contredit, compte tenu du fait que nous sommes dans le cas 3, la condition (V.1) ; donc $reg(\mathcal{B}_1|_{[Q_1,Q_2]}, \mathcal{B}_2|_{[Q_1,Q_2]})$ ne peut pas être convexe.

Soient $\mathcal{T}_2$ et $\mathcal{T}_2'$ les deux droites d'appui à $\mathcal{B}_2$ passant par $P_{m_1}^{(1)}$ et soient $\{T_2\} = \mathcal{T}_2 \cap \mathcal{B}_2$ et $\{T_2'\} = \mathcal{T}_2' \cap \mathcal{B}_2$. L'un des points T_2 et T_2' se trouve évidemment dans le demi-plan $\overline{DP_{op}(P_0^{(1)}P_{m_1}^{(1)}, \mathcal{P}_1)}$, soit T_2' celui-ci.

Un seul point d'intersection de $\mathcal{B}_1$ et $\mathcal{B}_2$ peut appartenir à $\mathcal{B}_2|_{[P_0^{(2)},T_2]}$, car sinon nous avons deux points d'intersection Q_1 et Q_2 de ces deux courbes pour lesquels

$$reg(\mathcal{B}_1|_{[Q_1,Q_2]}, \mathcal{B}_2|_{[Q_1,Q_2]})$$

est convexe, contradiction.

Symétriquement, soient $\mathcal{T}_1$ et $\mathcal{T}_1'$ les deux droites d'appui à $\mathcal{B}_1$ passant par $P_{m_2}^{(2)}$; T_1 et T_1' étant respectivement leurs points de contact à $\mathcal{B}_1$, l'un se trouvera dans le demi-plan $\overline{DP_{op}(P_0^{(2)}P_{m_2}^{(2)}, \mathcal{B}_2)}$ et nous considérons que c'est T_1'. Alors un seul point d'intersection de $\mathcal{B}_1$ et $\mathcal{B}_2$ peut appartenir à $\mathcal{B}_1|_{[P_0^{(1)},T_1]}$.

$\mathcal{B}_1 \cup [P_0^{(1)}P_{m_1}^{(1)}]$ et $\mathcal{B}_2 \cup [P_0^{(2)}P_{m_2}^{(2)}]$ sont des courbes fermées, et en plus $\mathcal{B}_1$ et $\mathcal{B}_2$ ne peuvent pas être tangentes l'une à l'autre, car l'arrangement qu'elles forment est simple. Par conséquent, $\mathrm{card}\big((\mathcal{B}_1 \cup [P_0^{(1)}P_{m_1}^{(1)}]) \cap (\mathcal{B}_2 \cup [P_0^{(2)}P_{m_2}^{(2)}])\big)$ est pair. Comme $\mathrm{card}([P_0^{(1)}P_{m_1}^{(1)}] \cap [P_0^{(2)}P_{m_2}^{(2)}]) = 1$, le cardinal de $\mathcal{B}_1 \cap \mathcal{B}_2$ doit être impair. En plus, nous avons supposé $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) > 1$, donc $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) \geq 3$ et soient Q_1, Q_2 et Q_3 trois points d'intersection des deux courbes distincts. En utilisant les observations antérieures, étudions cas par cas les placements possibles des points Q_1, Q_2 et Q_3 sur les courbes $\mathcal{B}_1$ et $\mathcal{B}_2$.

(a) $Q_1 \in \mathcal{B}_1|_{[P_0^{(1)},T_1]} \cap \mathcal{B}_2|_{[P_0^{(2)},T_2]}$
$Q_2, Q_3 \in \mathcal{B}_1|_{[T_1,P_{m_1}^{(1)}]} \cap \mathcal{B}_2|_{[T_2,P_{m_2}^{(2)}]}$

Nous aurons alors la situation de la figure V.6 (ou une symétrique en $\mathcal{B}_1$ et $\mathcal{B}_2$).

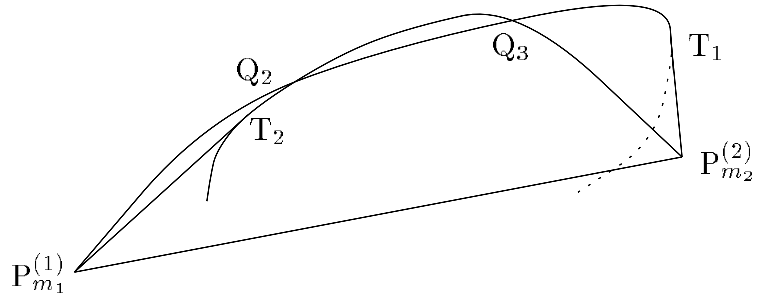


Figure V.6

Le polygone $\mathcal{P}_2$ doit alors être inclus dans le demi-plan $\overline{DP(P_{m_2}^{(2)}T_1, P_{m_1}^{(1)})}$ et il est très facile à remarquer que la condition (V.2) ne peut pas être satisfaite (en pointillés, une partie de $\mathcal{B}_1|_{[P_0^{(1)},T_1]}$).

La même contradiction s'obtient évidemment au cas où

$\quad Q_2, Q_3 \in \mathcal{B}_1\big|_{[T_1,P^{(1)}_{m_1}]} \cap \mathcal{B}_2\big|_{[T_2,P^{(2)}_{m_2}]}$ et

(b) $Q_1 \in \mathcal{B}_1\big|_{[T_1,P^{(1)}_{m_1}]} \cap \mathcal{B}_2\big|_{[T_2,P^{(2)}_{m_2}]}$

(c) $Q_1 \in \mathcal{B}_1\big|_{[T_1,P^{(1)}_{m_1}]} \cap \mathcal{B}_2\big|_{[P^{(2)}_0,T_2]}$

(d) $Q_1 \in \mathcal{B}_1\big|_{[P^{(1)}_0,T_1]} \cap \mathcal{B}_2\big|_{[T_2,P^{(2)}_{m_2}]}$

Il reste ainsi à étudier le cas

(e) $Q_1 \in \mathcal{B}_1\big|_{[P^{(1)}_0,T_1]} \cap \mathcal{B}_2\big|_{[T_2,P^{(2)}_{m_2}]}$

$\quad Q_2 \in \mathcal{B}_1\big|_{[T_1,P^{(1)}_{m_1}]} \cap \mathcal{B}_2\big|_{[P^{(2)}_0,T_2]}$

$\quad Q_3 \in \mathcal{B}_1\big|_{[T_1,P^{(1)}_{m_1}]} \cap \mathcal{B}_2\big|_{[T_2,P^{(2)}_{m_2}]}$

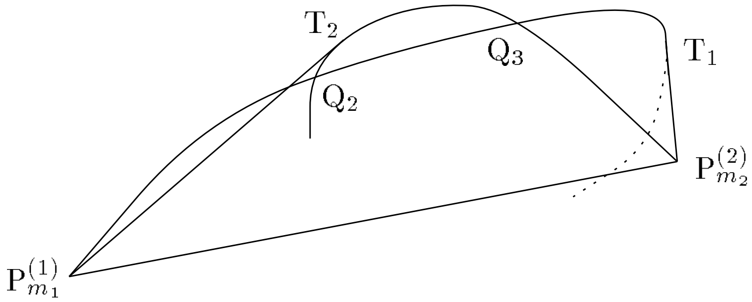


Figure V.7

Pour les points Q_2 et Q_3, la configuration est celle présentée par la figure V.7 et nous remarquons facilement que la même contradiction qu'au cas (a) peut être obtenue (même sans regarder la position du point Q_1).

Pour conclure, $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2)$ doit être inférieur à 3 et comme il est impair nous avons obtenu le résultat désiré et la preuve du lemme est achevée.

□

Les conditions (V.1), (V.2) et (V.3) sont basées sur les B-polygones des courbes de Bézier $\mathcal{B}_1$ et $\mathcal{B}_2$ et non pas sur les courbes elles-mêmes et par conséquent pourraient être utilisées dans notre algorithme pour assurer $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) = 1$. La question qui reste est si par subdivision de de Casteljau nous finirons par obtenir en un nombre fini d'itérations des courbes de Bézier satisfaisant ces trois conditions dans le cas où les deux courbes initiales, $\mathcal{B}_1$ et $\mathcal{B}_2$, ne les satisfaisaient pas. La réponse est malheureusement non : par exemple, si $\mathcal{B}_1$ et $\mathcal{B}_2$ se coupent en un point correspondant sur $\mathcal{B}_1$ au paramètre $t_1 = 1/2$ et sur $\mathcal{B}_2$ au paramètre $t_2 = 1/3$, après une subdivision de $\mathcal{B}_1$ nous retrouvons ce point d'intersection comme étant une extrémité commune de deux courbes de Bézier situées sur $\mathcal{B}_1$, tandis que sur $\mathcal{B}_2$ ce point sera toujours à l'intérieur du polygone de contrôle de la partie de courbe qui le contient (figure V.8), et par conséquent la condition (V.1) ne sera jamais satisfaite.

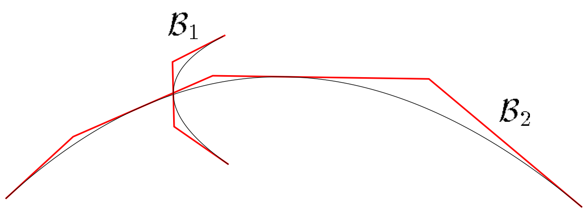


Figure V.8

Par conséquent, le lemme V.1 ne nous fournit pas un critère utilisable dans notre algorithme de calcul d'un arrangement de polygones de contrôle équivalent à un arrangement de polycourbes donné, car un tel algorithme ne convergerait pas toujours. Le lemme suivant est une généralisation de V.1 palliant ce problème.

Lemme V.2. – *Soient $\mathcal{B}_1$ et $\mathcal{B}_2$ deux courbes de Bézier complètement convexes telles que $\mathcal{A}(\{\mathcal{B}_1, \mathcal{B}_2\})$ soit un arrangement simple. Supposons que nous avons obtenu, par subdivision, une écriture de $\mathcal{B}_k$ sous la forme $\mathcal{B}_k = \cup_{i=1}^{n_k} \mathcal{B}_{k,i}$, pour $k = 1, 2$. Soient $\widetilde{\mathcal{P}}_k = \cup_{i=1}^{n_k} \mathcal{P}_k$, pour $k = 1, 2$, les deux polygones de contrôle courants et $\widetilde{\mathcal{S}}_k = \cup_{i=1}^{n_k} [\mathrm{P}_0^{(k,i)} \mathrm{P}_{m_{k,i}}^{(k,i)}]$, $k = 1, 2$, les deux polygones support courants des (poly)courbes $\mathcal{B}_1$ et $\mathcal{B}_2$. Si*

$$\mathrm{card}(\widetilde{\mathcal{P}}_1 \cap \widetilde{\mathcal{P}}_2) = \mathrm{card}(\widetilde{\mathcal{P}}_1 \cap \widetilde{\mathcal{S}}_2) = \mathrm{card}(\widetilde{\mathcal{S}}_1 \cap \widetilde{\mathcal{P}}_2) = \mathrm{card}(\widetilde{\mathcal{S}}_1 \cap \widetilde{\mathcal{S}}_2) = 1 \qquad \text{(V.4)}$$

et si $\widetilde{\mathcal{P}}_1$ et $\widetilde{\mathcal{P}}_2$ satisfont (V.2) et (V.3), alors $\mathrm{card}(\widetilde{\mathcal{B}}_1 \cap \widetilde{\mathcal{B}}_2) = 1$.

Preuve – La démonstration de ce lemme est en tout point semblable à celle du lemme V.1, en remplaçant $[\mathrm{P}_0^{(k)} \mathrm{P}_{m_k}^{(k)}]$ par $\widetilde{\mathcal{S}}_k$, pour $k = 1, 2$, et $reg(\mathcal{P}_k, [\mathrm{P}_0^{(k)} \mathrm{P}_{m_k}^{(k)}])$ par

$$\left(\bigcup_{i=1}^{n_k} reg(\mathcal{P}_{k,i}, [\mathrm{P}_0^{(k,i)} \mathrm{P}_{m_{k,i}}^{(k,i)}]) \right) \cup \{\mathrm{P}_0^{(k,2)}, \mathrm{P}_0^{(k,3)}, \ldots, \mathrm{P}_0^{(k,n_k)}\}$$

pour $k = 1, 2$, et en tenant compte de la convexité de $\widetilde{\mathcal{S}}_1$ et $\widetilde{\mathcal{S}}_2$. □

Pour pouvoir nous servir de ce résultat dans l'élaboration de notre algorithme, nous devons maintenant prouver que pour deux courbes de Bézier complètement convexes $\mathcal{B}_1$ et $\mathcal{B}_2$ il est possible d'obtenir, par subdivision de de Casteljau de rapport 1/2, des écritures satisfaisant les conditions (V.4), (V.2) et (V.3).

Hypothèse. Nous allons considérer à partir de maintenant et dans toute la suite du chapitre que si $\mathcal{B}_1$ et $\mathcal{B}_2$ sont deux courbes de Bézier, $\{\mathrm{P}_0^{(1)}, \mathrm{P}_{m_1}^{(1)}\} \cap \mathcal{B}_2 = \emptyset$ et $\{\mathrm{P}_0^{(2)}, \mathrm{P}_{m_2}^{(2)}\} \cap \mathcal{B}_1 = \emptyset$, car dans le cas contraire une situation similaire à celle de la figure V.8 (mais en considérant par exemple $\mathrm{B}_{m_1}(\mathrm{P}_0^{(1)}, \ldots, \mathrm{P}_{m_1}^{(1)}\,;\, 0) = \mathrm{B}_{m_2}(\mathrm{P}_0^{(2)}, \ldots, \mathrm{P}_{m_2}^{(2)}\,;\, 1/3)$ à la place de $\mathrm{B}_{m_1}(\mathrm{P}_0^{(1)}, \ldots, \mathrm{P}_{m_1}^{(1)}\,;\, 1/2) = \mathrm{B}_{m_2}(\mathrm{P}_0^{(2)}, \ldots, \mathrm{P}_{m_2}^{(2)}\,;\, 1/3)$) rendrait impossible l'obtention de polygones de contrôle/support pour les deux courbes satisfaisant la conditions (V.4).

Lemme V.3. – *Soient $\mathcal{B}_1$ et $\mathcal{B}_2$ deux courbes de Bézier complétement convexes telles que l'arrangement $\mathcal{A}(\{\mathcal{B}_1, \mathcal{B}_2\})$ soit simple et supposons que* $\mathrm{card}(\mathcal{B}_1 \cap \mathcal{B}_2) = 1$. *En notant* R *le point commun des deux courbes, nous pouvons obtenir par subdivision de de Casteljau de rapport 1/2 une écriture de $\mathcal{B}_k$ sous la forme $\mathcal{B}_k = \cup_{i=1}^{n_k} \mathcal{B}_{k,i}$, pour $k = 1, 2$, telles que si $i_1 \in \{1, \ldots, n_1\}$ et $i_2 \in \{1, \ldots, n_2\}$ ont la propriété $\mathrm{R} \in \mathcal{B}_{1,i_1} \cap \mathcal{B}_{2,i_2}$, alors il existe $j_k \in \{1, \ldots, i_k\}$ et $\ell_k \in \{i_k, \ldots, n_k\}$, $k = 1, 2$, tels qu'en mettant*

$$\overline{\mathcal{P}}_k = \bigcup_{i=j_k}^{\ell_k} \mathcal{P}_{k,i} \qquad et \qquad \overline{\mathcal{S}}_k = \bigcup_{i=j_k}^{\ell_k} [\mathrm{P}_0^{(k,i)} \mathrm{P}_{m_{k,i}}^{(k,i)}]\ , \quad k = 1, 2$$

les polygones $\overline{\mathcal{P}}_k$ et $\overline{\mathcal{S}}_k$, $k = 1, 2$, satisfassent (V.4) et $\overline{\mathcal{P}}_1$ et $\overline{\mathcal{P}}_2$ satisfassent (V.2) et (V.3).

Preuve – Remarquons que $m_{k,i} = m_k$ pour tout $i \in \{1, \ldots, n_k\}$, $k = 1, 2$, car toutes les courbes obtenues par la subdivision de $\mathcal{B}_k$ ont bien sûr le même degré que celle-ci. Nous convenons de noter $\mathrm{P}_0^{(k)}$ et $\mathrm{P}_{m_k}^{(k)}$ les extrémités de la courbe $\mathcal{B}_k$, car aucune confusion n'est possible.

De par l'hypothèse que nous avons fait, $\mathcal{B}_1 \cap \mathcal{B}_2 \subset \big(\mathcal{B}_1 \setminus \{\mathrm{P}_0^{(1)}, \mathrm{P}_{m_1}^{(1)}\}\big) \cap \big(\mathcal{B}_2 \setminus \{\mathrm{P}_0^{(2)}, \mathrm{P}_{m_2}^{(2)}\}\big)$; il est alors facile à voir qu'une démonstration similaire à celle du théorème IV.24 construit des écritures de $\mathcal{B}_1$ et $\mathcal{B}_2$ telles que les polygones $\widetilde{\mathcal{P}}_k$ et $\widetilde{\mathcal{S}}_k$, $k = 1, 2$, définis dans l'énoncé du lemme V.2 satisfassent (V.4), et donc il existe $j_k \in \{1, \ldots, i_k\}$ et $\ell_k \in \{i_k, \ldots, n_k\}$ tels que les polygones $\overline{\mathcal{P}}_k$ et $\overline{\mathcal{S}}_k$, $k = 1, 2$, satisfassent aussi cette condition.

Soit Δ la droite d'appui commune des courbes $\mathcal{B}_1$ et $\mathcal{B}_2$ ayant les propriétés (figure V.9)

$$\Delta \cap \mathcal{B}_1 \subset \mathrm{DP}_{\mathrm{op}}(\tan_{\mathrm{R}}(\mathcal{B}_2), \mathcal{B}_2) \qquad \text{et} \qquad \Delta \cap \mathcal{B}_2 \subset \mathrm{DP}_{\mathrm{op}}(\tan_{\mathrm{R}}(\mathcal{B}_1), \mathcal{B}_1)$$

Une telle droite d'appui existe, car $\mathcal{B}_1$ et $\mathcal{B}_2$ ne sont pas tangentes en R, et elle est unique (cette droite a déjà été utilisée dans la preuve du théorème IV.24). Nous notons $\{\mathrm{T}_1\} = \Delta \cap \mathcal{B}_1$ et $\{\mathrm{T}_2\} = \Delta \cap \mathcal{B}_2$ et soit P_1 le point de $\mathcal{B}_1$ qui divise en deux arcs égaux $\mathcal{B}_1\big|_{[\mathrm{R},\mathrm{T}_1]}$ et P_2 le point de $\mathcal{B}_2$ qui divise en deux arcs égaux $\mathcal{B}_2\big|_{[\mathrm{R},\mathrm{T}_2]}$. La tangente à $\mathcal{B}_1$ en P_1, $\tan_{\mathrm{P}_1}(\mathcal{B}_1)$, coupe l'arc $\mathcal{B}_2\big|_{]\mathrm{R},\mathrm{T}_2[}$ en un point Q_1 ; symétriquement, $\tan_{\mathrm{P}_2}(\mathcal{B}_2) \cap \mathcal{B}_1\big|_{]\mathrm{R},\mathrm{T}_1[} = \{\mathrm{Q}_2\}$. Pour fixer les notations, supposons que la valeur du paramètre est croissante sur $\mathcal{B}_1\big|_{[\mathrm{R},\mathrm{T}_1]}$ en allant de R vers T_1, et de même elle est croissante sur $\mathcal{B}_2\big|_{[\mathrm{R},\mathrm{T}_2]}$ en allant de R vers T_2 ; les extrémités des deux courbes seront alors dans la position de la figure V.9.

Il est évident que nous pouvons obtenir par subdivision une écriture de $\mathcal{B}_1$ telle qu'il existe un point $\mathrm{P}_0^{(1,\ell_1+1)}$ sur l'arc $\mathcal{B}_1\big|_{[\mathrm{Q}_1,\mathrm{P}_1]}$ et de même pour $\mathcal{B}_2$, nous pouvons obtenir par subdivision une écriture telle qu'il existe un point $\mathrm{P}_0^{(2,\ell_2+1)}$ sur l'arc $\mathcal{B}_2\big|_{[\mathrm{Q}_2,\mathrm{P}_2]}$. Soit $\{\mathrm{Q}_3\} = \mathrm{P}_1\mathrm{Q}_2 \cap \mathrm{P}_2 \cap \mathrm{Q}_1$; les positions des points $\mathrm{P}_0^{(1,\ell_1)}$ et $\mathrm{P}_0^{(2,\ell_2)}$ impliquent

$$\widetilde{\mathcal{P}}_1 \cap \widetilde{\mathcal{P}}_2 \in reg([\mathrm{Q}_1\mathrm{Q}_3], [\mathrm{Q}_3\mathrm{Q}_2], [\mathrm{Q}_2\mathrm{R}], [\mathrm{R}\mathrm{Q}_1])$$

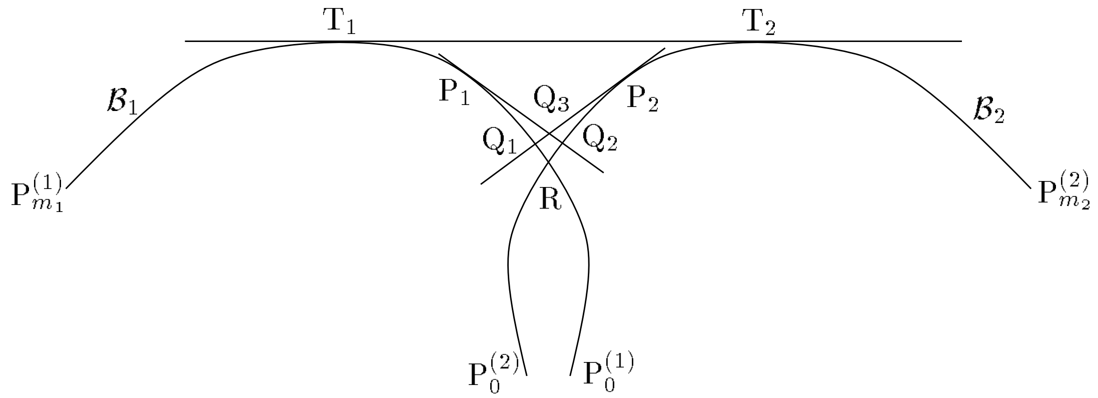


Figure V.9

En notant $\widetilde{\mathrm{P}}_{i_1}^{(1)}$, $\widetilde{\mathrm{P}}_{i_1+1}^{(1)}$, $\widetilde{\mathrm{P}}_{i_2}^{(2)}$ et $\widetilde{\mathrm{P}}_{i_2+1}^{(2)}$ les sommets de $\widetilde{\mathcal{P}}_1$, respectivement $\widetilde{\mathcal{P}}_2$, pour lesquels

$$[\widetilde{\mathrm{P}}_{i_1}^{(1)}\widetilde{\mathrm{P}}_{i_1+1}^{(1)}] \cap [\widetilde{\mathrm{P}}_{i_2}^{(2)}\widetilde{\mathrm{P}}_{i_2+1}^{(2)}] = \widetilde{\mathcal{P}}_1 \cap \widetilde{\mathcal{P}}_2 \ ,$$

la relation précédente nous dit que

$$\mathcal{B}_1\big|_{[\mathrm{Q}_1,\mathrm{P}_1]} \subset \mathrm{DP}_{\mathrm{op}}(\widetilde{\mathrm{P}}_{i_2}^{(2)}\widetilde{\mathrm{P}}_{i_2+1}^{(2)}, \mathcal{B}_2) \quad \text{et} \quad \mathcal{B}_2\big|_{[\mathrm{Q}_2,\mathrm{P}_2]} \subset \mathrm{DP}_{\mathrm{op}}(\widetilde{\mathrm{P}}_{i_1}^{(1)}\widetilde{\mathrm{P}}_{i_1+1}^{(1)}, \mathcal{B}_1)$$

Comme $\mathrm{P}_0^{(k,\ell_k+1)} = \mathrm{P}_{m_{k,\ell}}^{(k,\ell_k)}$ et les indices des sommets de $\widetilde{\mathcal{P}}_k$ sont bien sûr croissants de $\mathrm{P}_0^{(k)}$ vers $\mathrm{P}_{m_k}^{(k)}$, pour $k = 1, 2$, il suffit de prendre $j_k = 1$ pour $k = 1, 2$ pour que toutes les conditions du lemme soient satisfaites. □

V.3.2. *Les triangles simples*

Soit $\mathcal{B} = \{\mathcal{B}_i\}_{i=1}^n$ une famille de polcourbes de Bézier telle que l'arrangement $\mathcal{A}(\mathcal{B})$ soit simple et supposons que dans son écriture initiale toutes deux courbes de Bézier n'appartenant pas à la même polycourbe se coupent en au plus un point. Cette supposition ne restreint pas la généralité, car nous avons vu au début de la preuve du théorème IV.24 que pour un arrangement ne la satisfaisant pas nous pouvons obtenir par subdivision une écriture qui la satisfait, et nous n'avons qu'à considérer cette nouvelle écriture comme étant celle initiale de l'arrangement.

Les résultats de la sous-section précédente nous permettent de concevoir un algorithme qui construit, par subdivisions de de Casteljau successives, des écritures des polycourbes $\mathcal{B}_i$, pour tout $i \in \{1, \ldots, n\}$, comme $\mathcal{B}_i = \cup_{j=1}^{n_i} \mathcal{B}_{i,j}$ telles que

$$\begin{gathered} \operatorname{card}(\widetilde{\mathcal{B}}_{i_1,j_1} \cap \widetilde{\mathcal{B}}_{i_2,j_2}) = \operatorname{card}(\widetilde{\mathcal{P}}_{i_1,j_1} \cap \widetilde{\mathcal{P}}_{i_2,j_2}) = \operatorname{card}(\widetilde{\mathcal{S}}_{i_1,j_1} \cap \widetilde{\mathcal{S}}_{i_2,j_2}) \in \{0,1\} \\ \forall i_1 \neq i_2 \in \{1, \ldots, n\} \forall j_1 \in \{1, \ldots, n_{i_1}\} \forall j_2 \in \{1, \ldots, n_{i_2}\} \end{gathered} \tag{V.5}$$

Mais cette condition n'est pas suffisante pour l'équivalence des arrangements $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$. Par exemple, la figure V.10 nous présente un arrangement de trois polycourbes de Bézier composées chacune d'une seule courbe de Bézier qui satisfont la relation (V.5) sans que les deux arrangements soient équivalents.

Nous allons prouver que la condition (V.5) est suffisante pour l'équivalence des deux arrangements, à l'exception des cas où des cellules comme celles décrites par la définition suivante existent dans l'arrangement $\mathcal{A}(\mathcal{B})$:

Définition V.4. – *Soit $\mathcal{B}$ un ensemble de polycourbes de Bézier tel que l'arrangement $\mathcal{A}(\mathcal{B})$ soit simple. Nous appelons triangle simple tout triplet $\big((i_1, j_1), (i_2, j_2), (i_3, j_3)\big)$ avec $i_k \in \{1, \ldots, n\}$ tous distincts et $j_k \in \{1, \ldots, n_{i_k}\}$ pour $k = 1, 2, 3$ tel que*

$$\widetilde{\mathcal{B}}_{i_1,j_1} \cap \widetilde{\mathcal{B}}_{i_2,j_2} \neq \emptyset \quad \widetilde{\mathcal{B}}_{i_1,j_1} \cap \widetilde{\mathcal{B}}_{i_3,j_3} \neq \emptyset \quad \widetilde{\mathcal{B}}_{i_2,j_2} \cap \widetilde{\mathcal{B}}_{i_3,j_3} \neq \emptyset$$

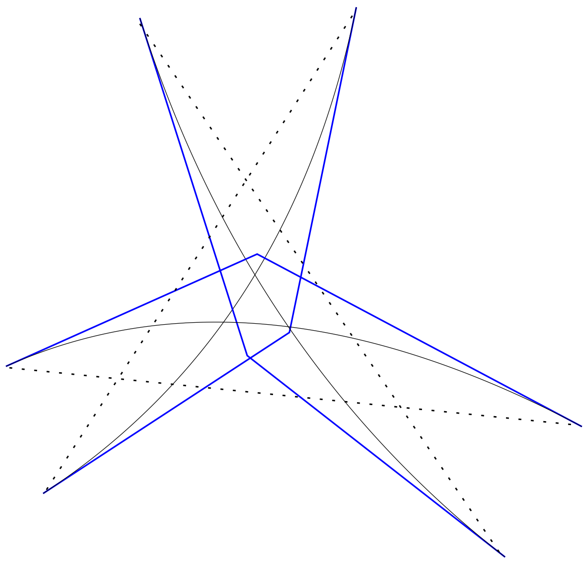


Figure V.10

Si la condition (V.5) est satisfaite, ce que nous allons supposer dans tout le reste de cette section, alors les trois courbes de Bézier désignées par un triangle simple forment effectivement un unique pseudo-triangle.

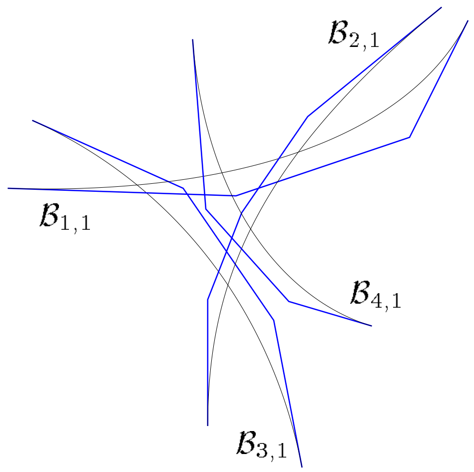


Figure V.11 *Le triplet* $\big((1,1),(2,1),(3,1)\big)$ *est un triangle simple sans définir une cellule de* $\mathcal{A}(\mathcal{B})$.

Remarque. Un triangle simple ne représente pas forcément une cellule de $\mathcal{A}(\mathcal{B})$, comme nous pouvons le voir dans la figure V.11.

Notation. Si $\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ est un triangle simple de $\mathcal{A}(\mathcal{B})$, nous allons l'appeler $\mathcal{T}_{\mathcal{B}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ pour identifier quel est l'arrangement auquel il fait référence.

Nous pouvons facilement voir que si nous avons, dans l'arrangement $\mathcal{A}(\mathcal{B})$, un triangle simple $\mathcal{T}_{\mathcal{B}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ et si la condition (V.5) est satisfaite, alors les B-polygones $\mathcal{P}_{i_1,j_1}$, $\mathcal{P}_{i_2,j_2}$ et $\mathcal{P}_{i_3,j_3}$ définissent dans $\mathcal{A}(\mathcal{P})$ un unique triangle simple, que nous noterons $\mathcal{T}_{\mathcal{P}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$. Nous observons que la simplicité de l'arrangement $\mathcal{A}(\mathcal{B})$ n'implique pas celle de $\mathcal{A}(\mathcal{P})$, même si la condition (V.5) est vérifiée, et les trois polygones de contrôle peuvent avoir un point commun, cas auquel nous dirons que le triangle $\mathcal{T}_{\mathcal{P}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ est dégénéré.

Définition V.5. – *Soit* $\mathcal{T}_{\mathcal{B}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ *un triangle simple de l'arrangement* $\mathcal{A}(\mathcal{B})$. *Nous dirons que le triangle simple* $\mathcal{T}_{\mathcal{P}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ *de l'arrangement* $\mathcal{A}(\mathcal{P})$ *lui est équivalent ssi* $\mathcal{T}_{\mathcal{P}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ *n'est pas dégénéré et l'ordre des points* $\widetilde{\mathcal{B}}_{i_1,j_1}\cap\widetilde{\mathcal{B}}_{i_2,j_2}$ *et* $\widetilde{\mathcal{B}}_{i_3,j_3}\cap\widetilde{\mathcal{B}}_{i_1,j_1}$ *sur* $\widetilde{\mathcal{B}}_{i_1,j_1}$ *est le même que l'ordre des points* $\widetilde{\mathcal{P}}_{i_1,j_1}\cap\widetilde{\mathcal{P}}_{i_2,j_2}$ *et* $\widetilde{\mathcal{P}}_{i_3,j_3}\cap\widetilde{\mathcal{P}}_{i_1,j_1}$ *sur* $\widetilde{\mathcal{P}}_{i_1,j_1}$.

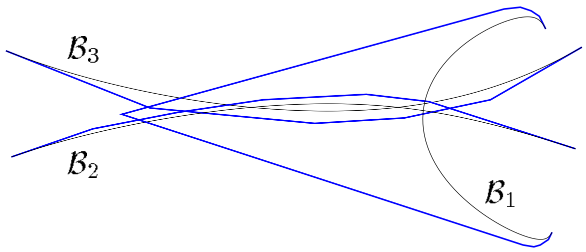


Figure V.12 *Si* $\mathcal{B}_2,\mathcal{P}_2$ *et* $\mathcal{B}_3,\mathcal{P}_3$ *sont parcourus de gauche à droite et* $\mathcal{B}_1,\mathcal{P}_1$ *de haut en bas,* $\mathcal{P}_1$ *coupe d'abord* $\mathcal{P}_3$ *et ensuite* $\mathcal{P}_2$ *et* $\mathcal{B}_1$ *coupe d'abord* $\mathcal{B}_3$ *et ensuite* $\mathcal{B}_2$. *Mais* $\mathcal{P}_2$ *coupe d'abord* $\mathcal{P}_1$, *ensuite* $\mathcal{P}_2$ *deux fois, tandis que* $\mathcal{B}_2$ *coupe d'abord* $\mathcal{B}_2$ *deux fois et seulement ensuite* $\mathcal{B}_1$.

Évidemment, si $\mathcal{T}_{\mathcal{P}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ est équivalent à $\mathcal{T}_{\mathcal{B}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$, alors la condition (V.5) implique que l'ordre des points $\widetilde{\mathcal{B}}_{i_2,j_2} \cap \widetilde{\mathcal{B}}_{i_3,j_3}$ et $\widetilde{\mathcal{B}}_{i_1,j_1} \cap \widetilde{\mathcal{B}}_{i_2,j_2}$ sur $\widetilde{\mathcal{B}}_{i_2,j_2}$ est le même que l'ordre des points $\widetilde{\mathcal{P}}_{i_2,j_2} \cap \widetilde{\mathcal{P}}_{i_3,j_3}$ et $\widetilde{\mathcal{P}}_{i_1,j_1} \cap \widetilde{\mathcal{P}}_{i_2,j_2}$ sur $\widetilde{\mathcal{P}}_{i_2,j_2}$ et l'ordre des points $\widetilde{\mathcal{B}}_{i_3,j_3} \cap \widetilde{\mathcal{B}}_{i_1,j_1}$ et $\widetilde{\mathcal{B}}_{i_2,j_2} \cap \widetilde{\mathcal{B}}_{i_3,j_3}$ sur $\widetilde{\mathcal{B}}_{i_3,j_3}$ est le même que l'ordre des points $\widetilde{\mathcal{P}}_{i_3,j_3} \cap \widetilde{\mathcal{P}}_{i_1,j_1}$ et $\widetilde{\mathcal{P}}_{i_2,j_2} \cap \widetilde{\mathcal{P}}_{i_3,j_3}$ sur $\widetilde{\mathcal{P}}_{i_3,j_3}$. La figure suivante donne un exemple où la relation (V.5) n'est pas satisfaite et où l'affirmation précédente est fausse.

En utilisant les notations et définitions données, nous pouvons formuler des conditions suffisantes d'équivalence des arrangements $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$:

Lemme V.6. – *Soit $\mathcal{B} = \{\mathcal{B}_i\}_{i=1}^n$ une famille de polcourbes de Bézier telle que l'arrangement $\mathcal{A}(\mathcal{B})$ soit simple et que la condition (V.5) soit satisfaite. Supposons que pour tout triangle simple $\mathcal{T}_{\mathcal{B}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ le triangle simple $\mathcal{T}_{\mathcal{P}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ de $\mathcal{A}(\mathcal{P})$ lui est équivalent. Alors les arrangements $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$ sont équivalents.*

Preuve – Nous allons montrer que si les propriétés de l'énoncé sont satisfaites, alors les conditions de la proposition IV.23 sont vérifiées par les arrangements $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$ et par conséquent ces deux arrangements sont équivalents.

Comme nous avons déjà mentionné à la fin de la preuve du théorème IV.24, les propriétés des courbes de Bézier complètement convexes font que les arrangements $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$ satisfont les conditions (IV.23.i) et (IV.23.ii) ; il nous reste donc à étudier les hypothèses (h1) et (h2).

Le fait que $\mathcal{A}(\mathcal{P})$ est simple résulte de l'équivalence des triangles simples de $\mathcal{A}(\mathcal{P})$ avec les triangles simples leur correspondant dans $\mathcal{A}(\mathcal{B})$. En effet, si nous supposons que l'arrangement des polygones de contrôle n'est pas simple, il existe un triplet de polygones de contrôle $\mathcal{P}_{i_1,j_1}$, $\mathcal{P}_{i_2,j_2}$ et $\mathcal{P}_{i_3,j_3}$ qui ont un point commun ; mais alors les polygones $\widetilde{\mathcal{P}}_{i_1,j_1}$, $\widetilde{\mathcal{P}}_{i_2,j_2}$ et $\widetilde{\mathcal{P}}_{i_3,j_3}$ ont un point commun et courbes $\widetilde{\mathcal{B}}_{i_1,j_1}$, $\widetilde{\mathcal{B}}_{i_2,j_2}$ et $\widetilde{\mathcal{B}}_{i_3,j_3}$ devront elles aussi en avoir un, de par l'équivalence de $\mathcal{T}_{\mathcal{P}}\big((i_1,j_i),(i_2,j_2),(i_3,j_3)\big)$ et $\mathcal{T}_{\mathcal{B}}\big((i_1,j_i),(i_2,j_2),(i_3,j_3)\big)$, ce qui contredit la simplicité de $\mathcal{A}(\mathcal{B})$.

La bijection φ_0 de l'hypothèse (h2) est très facile à construire : tout sommet s de $\mathcal{A}(\mathcal{B})$ est l'intersection de deux courbes $\widetilde{\mathcal{B}}_{i_1,j_1}$ et $\widetilde{\mathcal{B}}_{i_2,j_2}$ et nous lui faisons correspondre le sommet de $\mathcal{A}(\mathcal{P})$ qui est l'intersection de $\widetilde{\mathcal{P}}_{i_1,j_1}$ et $\widetilde{\mathcal{P}}_{i_2,j_2}$; la propriété (V.5) nous dit que ce sommet est bien défini, le point d'intersection de $\widetilde{\mathcal{P}}_{i_1,j_1}$ et $\widetilde{\mathcal{P}}_{i_2,j_2}$ existant et étant unique. La condition (h2.1) est alors satisfaite par définition et nous devons étudier la propriété (h2.2).

Supposons que cette propriété n'est pas satisfaite ; alors il existe (i_1,j_1), (i_1,j_1'), (i_2,j_2) et (i_3,j_3) tels que l'ordre des sommets $\widetilde{\mathcal{B}}_{i_1,j_1} \cap \widetilde{\mathcal{B}}_{i_2,j_2}$ et $\widetilde{\mathcal{B}}_{i_1,j_1'} \cap \widetilde{\mathcal{B}}_{i_3,j_3}$ sur la polycourbe $\mathcal{B}_{i_1}$ ne soit pas le même que l'ordre des sommets $\widetilde{\mathcal{P}}_{i_1,j_1} \cap \widetilde{\mathcal{P}}_{i_2,j_2}$ et $\widetilde{\mathcal{P}}_{i_1,j_1'} \cap \widetilde{\mathcal{P}}_{i_3,j_3}$ sur le polygone $\mathcal{P}_{i_1}$.

Mais j_1 et j_1' ne peuvent pas être différents, car dans ce cas nous obtiendrons $j_1 < j_1'$ et $j_1' < j_1$, absurde. D'un autre côté, les courbes $\widetilde{\mathcal{B}}_{i_2,j_2}$ et $\widetilde{\mathcal{B}}_{i_3,j_3}$ doivent se couper, car si nous supposons le contraire, alors $\widetilde{\mathcal{P}}_{i_2,j_2} \cap \widetilde{\mathcal{P}}_{i_3,j_3} = \emptyset$ et par conséquent il existe une ligne polygonale non-bornée Δ, composée d'au plus deux demi-droites telle que $\widetilde{\mathcal{P}}_{i_2,j_2}$ et $\widetilde{\mathcal{P}}_{i_3,j_3}$ soient chacun inclus dans une des deux parties connexes ouvertes définies par Δ et telle que $\mathrm{card}(\Delta \cap \widetilde{\mathcal{S}}_{i_1,j_1}) = 1$, car la relation (V.5) est vérifiée.

Dans ce cas, la supposition faite sur les ordres des sommets $\widetilde{\mathcal{B}}_{i_1,j_1} \cap \widetilde{\mathcal{B}}_{i_2,j_2}$ et $\widetilde{\mathcal{B}}_{i_1,j_1} \cap \widetilde{\mathcal{B}}_{i_3,j_3}$

sur la polycourbe $\mathcal{B}_{i_1}$, respectivement l'ordre des sommets $\widetilde{\mathcal{P}}_{i_1,j_1} \cap \widetilde{\mathcal{P}}_{i_2,j_2}$ et $\widetilde{\mathcal{P}}_{i_1,j_1} \cap \widetilde{\mathcal{P}}_{i_3,j_3}$ sur le polygone $\mathcal{P}_{i_1}$, signifie que tandis que la partie de la courbe $\widetilde{\mathcal{B}}_{i_1,j_1}$ comprise entre les deux sommets traverse Δ par exemple de gauche à droite, la partie de son polygone associé $\widetilde{\mathcal{P}}_{i_1,j_1}$ comprise respectivement entre les deux sommets la traverse de droite à gauche, ce qui est évidemment impossible dans le cas des courbes de Bézier complètement convexes.

Nous en déduisons que les courbes $\widetilde{\mathcal{B}}_{i_2,j_2}$ et $\widetilde{\mathcal{B}}_{i_3,j_3}$ ne sont pas disjointes. Dans ce cas, $(i_1, j_1 = j_i')$, (i_2, j_2) et (i_3, j_3) définissent un triangle simple dans l'arrangement $\mathcal{A}(\mathcal{B})$ et, suite à la condition (V.5), un autre dans l'arrangement $\mathcal{A}(\mathcal{P})$; l'inversion des ordres des sommets sur $\widetilde{\mathcal{B}}_{i_1,j_1}$, respectivement $\widetilde{\mathcal{P}}_{i_1,j_1}$ contredit alors la condition d'équivalence des triangles simples des deux arrangements et ainsi la démonstration du lemme est achevée.

$\square$

V.3.3. *Équivalence de triangles simples*

En nous basant sur le lemme précédent, il nous suffit de trouver un critère d'équivalence des triangles simples $\mathcal{T}_{\mathcal{B}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$et $\mathcal{T}_{\mathcal{P}}\big((i_1,j_1),(i_2,j_2),(i_3,j_3)\big)$ utilisant uniquement les polygones associés aux courbes de $\widetilde{\mathcal{B}}_{i_1,j_1}$, $\widetilde{\mathcal{B}}_{i_2,j_2}$ et $\widetilde{\mathcal{B}}_{i_3,j_3}$ pour pouvoir concévoir l'algorithme désiré de calcul d'une famille $\mathcal{P}$ de polygones de contrôle telle que $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$ soient équivalents.

Remarque. Comme nous avons déjà mentionné, si la condition (V.5) est vérifiée alors les trois courbes de Bézier identifiées par un triangle simple de $\mathcal{A}(\mathcal{B})$ définissent un unique pseudo-triangle, qui peut ne pas être une cellule de l'arrangement. Dans la suite de cette sous-section, les deux notions (triplet de courbes et pseudo-triangle) seront confondues, la distinction se faisant d'après le contexte.

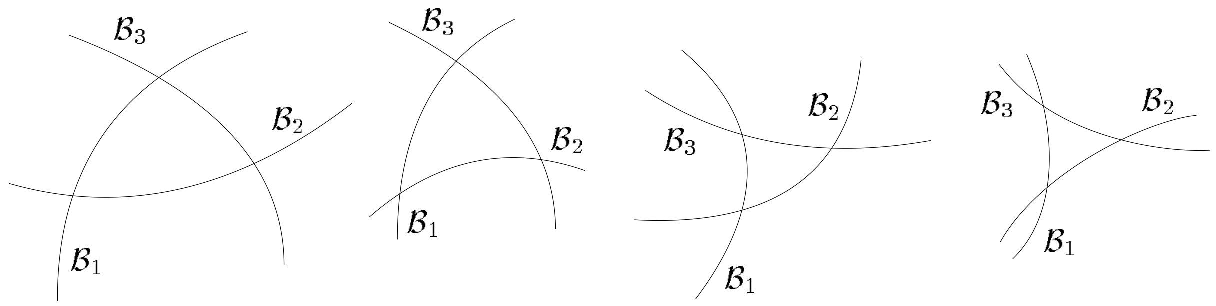


Figure V.13

Soient $\mathcal{B}_1$, $\mathcal{B}_2$ et $\mathcal{B}_3$ trois courbes de Bézier complétement convexes telles que l'arrangement $\mathcal{A}(\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3)$ soit simple et satisfasse (V.2), (V.3) et (V.4) (par conséquent, il satisfait (V.5)). $\mathcal{B}_1$, $\mathcal{B}_2$ et $\mathcal{B}_3$ peuvent définir quatre types de triangles simples :

T1. $\mathrm{Int}(\mathcal{T}_{\mathcal{B}}(1,2,3) \cap reg(\mathcal{B}_k, [\mathrm{P}_0^{(k)}\mathrm{P}_{m_k}^{(k)}]) \neq \emptyset$ pour tout $k \in \{1,2,3\}$. Nous appelerons un tel triangle simple triangle (3-convexe, 0-concave) ;

T2. il existe un unique $k_0 \in \{1,2,3\}$ tel que $\mathrm{Int}(\mathcal{T}_{\mathcal{B}}(1,2,3) \cap reg(\mathcal{B}_{k_0}, [\mathrm{P}_0^{(k_0)}\mathrm{P}_{m_{k_0}}^{(k_0)}]) = \emptyset$ et $\mathrm{Int}(\mathcal{T}_{\mathcal{B}}(1,2,3) \cap reg(\mathcal{B}_k, [\mathrm{P}_0^{(k)}\mathrm{P}_{m_k}^{(k)}]) \neq \emptyset$ pour tout $k \neq k_0 \in \{1,2,3\}$. Un tel triangle simple s'appelera triangle (2-convexe, 1-concave) ;

T3. il existe un unique $k_0 \in \{1,2,3\}$ tel que $\mathrm{Int}(\mathcal{T}_{\mathcal{B}}(1,2,3) \cap reg(\mathcal{B}_{k_0}, [\mathrm{P}_0^{(k_0)}\mathrm{P}_{m_{k_0}}^{(k_0)}]) \neq \emptyset$ et $\mathrm{Int}(\mathcal{T}_{\mathcal{B}}(1,2,3) \cap reg(\mathcal{B}_k, [\mathrm{P}_0^{(k)}\mathrm{P}_{m_k}^{(k)}]) = \emptyset$ pour tout $k \neq k_0 \in \{1,2,3\}$. Un tel triangle simple s'appelera triangle (1-convexe, 3-concave) ;

T4. $\mathrm{Int}(\mathcal{T}_{\mathcal{B}}(1,2,3) \cap reg(\mathcal{B}_k, [\mathrm{P}_0^{(k)}\mathrm{P}_{m_k}^{(k)}]) = \emptyset$ pour tout $k \in \{1,2,3\}$. Nous appelerons un tel triangle simple triangle (0-convexe, 3-concave).

La figure V.13 présente des exemples pour chacun des types de triangles simples énumérés.

De manière tout-à-fait analogue nous définitssons les triangles (i-convexes, j-concaves) pour $i, j \geq 0$ et tels que $i + j = 3$ pour $\mathcal{T}_{\mathcal{P}}(1,2,3)$. Regardons maintenant quels peuvent être les types du triangle $\mathcal{T}_{\mathcal{P}}(1,2,3)$ si nous connaissons le type du triangle $\mathcal{T}_{\mathcal{B}}(1,2,3)$.

- Si $\mathcal{T}_{\mathcal{B}}(1,2,3)$ est (3-convexe, 0-concave), $\mathcal{T}_{\mathcal{P}}(1,2,3)$ ne peut être que (3-convexe, 0-concave) et équivalent à $\mathcal{T}_{\mathcal{B}}(1,2,3)$.
- Si $\mathcal{T}_{\mathcal{B}}(1,2,3)$ est (2-convexe, 1-concave), $\mathcal{T}_{\mathcal{P}}(1,2,3)$ peut être soit (2-convexe, 1-concave), cas où il est équivalent à $\mathcal{T}_{\mathcal{B}}(1,2,3)$, soit (1-convexe, 2-concave), voir figure V.14(a).
- Si $\mathcal{T}_{\mathcal{B}}(1,2,3)$ est (1-convexe, 2-concave), $\mathcal{T}_{\mathcal{P}}(1,2,3)$ peut être soit (1-convexe, 2-concave), cas où il est équivalent à $\mathcal{T}_{\mathcal{B}}(1,2,3)$, soit (2-convexe, 1-concave), voir figure V.14(b).
- Si $\mathcal{T}_{\mathcal{B}}(1,2,3)$ est (0-convexe, 3-concave), $\mathcal{T}_{\mathcal{P}}(1,2,3)$ peut être soit (0-convexe, 3-concave), cas où il est équivalent à $\mathcal{T}_{\mathcal{B}}(1,2,3)$, soit (3-convexe, 0-concave), qui est le cas de la figure V.10.

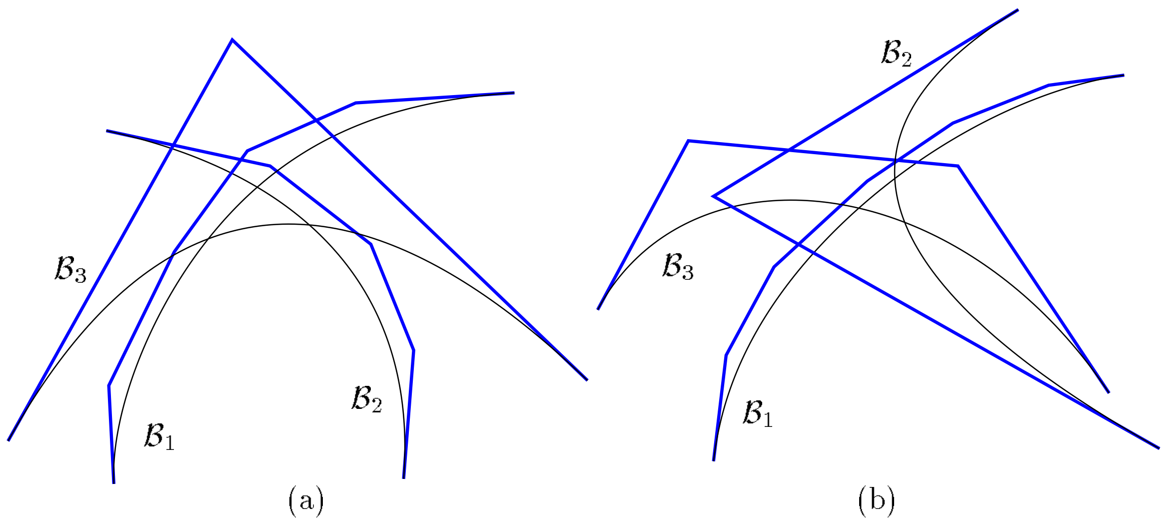


Figure V.14

Soit $\mathcal{S} = \{[\mathrm{P}_0^{(k)}\mathrm{P}_{m_k}^{(k)}]\}_{k=1,2,3}$ l'ensemble des *segments support* des courbes de Bézier $\mathcal{B}_k$, pour $k = 1,2,3$. Une étude de cas facile à faire nous dit que si $\mathcal{T}_{\mathcal{P}}(1,2,3)$ est équivalent à $\mathcal{T}_{\mathcal{S}}(1,2,3)$, alors il est aussi équivalent à $\mathcal{T}_{\mathcal{B}}(1,2,3)$. Ainsi, nous pouvons décider si les triangles $\mathcal{T}_{\mathcal{P}}(1,2,3)$ et $\mathcal{T}_{\mathcal{B}}(1,2,3)$ sont équivalents en regardant uniquement les ensembles de lignes polygonales $\mathcal{P}$ et $\mathcal{S}$ et par conséquent le critère d'équivalence de ces deux triangles que nous cherchions est trouvé.

Si les triangles $\mathcal{T}_{\mathcal{P}}(1,2,3)$ et $\mathcal{T}_{\mathcal{S}}(1,2,3)$ sont équivalents, nous savons que $\mathcal{T}_{\mathcal{P}}(1,2,3)$ est équivalent à $\mathcal{T}_{\mathcal{B}}(1,2,3)$ et le problème est résolu. Mais dans le cas contraire, il ne suffit pas de dire que nous ne pouvons pas affirmer l'équivalence des triangles $\mathcal{T}_{\mathcal{P}}(1,2,3)$ et $\mathcal{T}_{\mathcal{B}}(1,2,3)$, nous devons subdiviser les courbes $\mathcal{B}_k$, $k = 1,2,3$, jusqu'à l'obtention de la propriété désirée. Après σ subdivisions, supposons que la courbe de Bézier $\mathcal{B}_k$ s'écrit $\mathcal{B}_k = \cup_{i=1}^{n_k}\mathcal{B}_{k,i}$, pour $k = 1,2,3$. Alors $\mathcal{P}$ sera

$$\mathcal{P} = \left\{\bigcup_{i=1}^{n_k}\mathcal{P}_{k,i}\right\}_{k=1,2,3}$$

et $\mathcal{S}$ sera

$$\mathcal{S} = \left\{ \bigcup_{i=1}^{n_k} [\mathrm{P}_0^{(k,i)} \mathrm{P}_{m_k}^{(k,i)}] \right\}_{k=1,2,3}$$

et pour avoir l'équivalence des triangles $\mathcal{T}_{\mathcal{P}}(1,2,3)$ et $\mathcal{T}_{\mathcal{B}}(1,2,3)$ il suffira toujours que $\mathcal{T}_{\mathcal{P}}(1,2,3)$ et $\mathcal{T}_{\mathcal{S}}(1,2,3)$ soient équivalentes. Comme $\mathcal{P}_k$ et $\mathcal{S}_k$ convergent, par subdivision de de Casteljau de rapport 1/2, vers $\mathcal{B}_k$, pour $k = 1, 2, 3$, après un nombre fini d'itérations l'équivalence de ces deux derniers triangles sera vérifiée. Par exemple, en nous référant à la figure V.14(a) pour un triangle $\mathcal{T}_{\mathcal{B}}(1,2,3)$ (2-convex, 1-concave), il suffira de subdiviser jusqu'à ce que

$$\delta^{\mathrm{H}}(\widetilde{\mathcal{P}}_3, \widetilde{\mathcal{B}}_3) < \mathrm{d}(\widetilde{\mathcal{B}}_1 \cap \widetilde{\mathcal{B}}_2, \widetilde{\mathcal{B}}_3)$$

(nous utilisons bien sûr les notations de T2 pour $k_0 = 3$).

Corollaire V.7. – *Si au moment ou l'arrangement des polygones de contrôle est équivalent à celui de polycourbes toutes les lignes polygonales support sont simples, alors l'arrangement des lignes polygonales support est équivalent aux deux autres arrangements.*

Cette affirmation résulte immédiatement de la relation (V.5) et du critère décrit ci-dessus que nous utiliserons pour l'équivalence des triangles simples de $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$. La condition de simplicité des lignes polygonales support est nécessaire à cause de l'existence des configurations comme celle de la figure suivante, où la courbe de Bézier (qui constitue à elle seule une polycourbe dans ce cas) $\widetilde{\mathcal{B}}_1$ a comme polygone support $\mathcal{S}_1 = [\mathrm{PQ}] \cup [\mathrm{QP}]$.

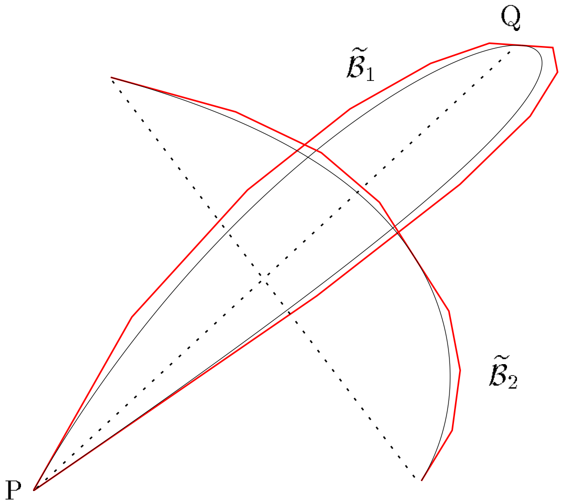


Figure V.15

Bien sûr, si une telle situation apparaît, elle est très facile à éliminer par la subdivision d'une des deux courbes de Bézier de l'écriture courante de $\widetilde{\mathcal{B}}_1$. Par conséquent, nous disposons de deux arrangements de lignes polygonales équivalents à $\mathcal{A}(\mathcal{B})$, et l'arrangement de lignes support est de taille évidemment inférieure à celle de l'arrangement de polygones de contrôle.

V.4. Présentation de l'algorithme

En nous basant sur les résultats de la section précédente, nous allons présenter dans cette section un algorithme calculant, pour une famille de polycourbes de Bézier donnée telle que l'arrangement $\mathcal{A}(\mathcal{B})$ soit simple, une famille de polygones de contrôle $\mathcal{P}$ telle que les arrangements $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$ soient équivalents.

V.4.1. *Structures de données utilisées*

Bien sûr, nous avons toujours besoin de la classe `courbe_de_Bézier` ; par contre, si une classe codant une polycourbe nous est aussi nécessaire, celle introduite dans la sous-section III.3.1, `polycourbe`, n'est pas adaptée aux procédures de ce chapitre. Nous la remplaçons par la classe `tab_polycourbe`, qui est en fait `méta_polygone<courbe_de_Bézier>`. Par un léger abus de notation, nous appellerons `arrangement` la classe `méta_polygone<tab_polycourbe>`, qui n'est donc en fait pas un arrangement, mais une famille de polycourbes, et celle-ci sera la classe de base de notre algorithme.

Si $\mathcal{A}$ est un `arrangement`, $\mathcal{A}[i]$ sera une `tab_polycourbe` et $\mathcal{A}[i][j]$ sera une `courbe_de_Bézier`. Pour simplifier les notations, nous introduisons une nouvelle classe, `ident_courbe`, ayant deux champs :

- `quelle_polycourbe`, un entier qui donnera l'indice dans l'`arrangement` de la polycourbe à laquelle la courbe de Bézier appartient ;
- `quelle_courbe`, un entier qui donnera l'indice de la courbe de Bézier dans $\mathcal{A}$[`quelle_polycourbe`].

Ainsi, si `id` est un `ident_courbe`, $\mathcal{A}$(`id`) sera $\mathcal{A}$[`id.quelle_polycourbe`][`id.quelle_courbe`].

Deux autres classes qui nous seront nécessaires sont `paire_de_courbes`, qui a deux champs, `première_courbe` et `seconde_courbe`, chacun étant un `ident_courbe`, et `triangle_simple`, qui a trois champs, `première_courbe`, `seconde_courbe` et `troisième_courbe`, chacun étant aussi un `ident_courbe`.

Très schématiquement, l'algorithme que nous proposons consiste à construire la liste de toutes les paires de courbes de Bézier intervenant dans l'arrangement qui ne satisfont pas (V.5), et à subdiviser toutes les courbes apparaissant dans cette liste, dans une première phase, et ensuite de traiter les triangles simples. Nous verrons dans les descriptions des procédures que l'algorithme appelle que la structure de données codant la liste de paires de courbes doit permettre d'effectuer de manière optimale l'ensemble d'opérations suivantes :

- insertion ordonnée d'une nouvelle `paire_de_courbes` (en utilisant l'ordre lexicographique pour cette classe) ;
- suppression d'une `paire_de_courbes` ;
- recherche/localisation d'une `paire_de_courbes` ;
- successeur ;
- prédécesseur.

Cette structure de données sera donc un dictionnaire augmenté ; en l'implémentant à l'aide d'un arbre rouge-noir, l'insertion, a suppression et la recherche d'une `paire_de_courbes` se fait en $\mathcal{O}(\log k)$, où k est la taille du dictionnaire, et les deux dernières opérations de la liste ci-dessus se font en temps constant (pour plus de détails sur ces structures de données, voir [Boi95] chap. 2).

Nous aurons aussi besoin d'une structure codant la liste des courbes qui doivent être subdivisées à chaque itération, et les opérations qui seront exécutées sur cette structure sont les mêmes que dans le cas des `paires_de_courbes` ; ainsi, un `dico_paires_de_courbes` est un dictionnaire augmenté de `paires_de_courbes`, un `dico_courbes` est un dictionnaire augmenté de `ident_courbe` ; en plus, nous introduisons le `tableau_triangles`, un tableau de `triangles_simples`.

Enfin, nous appelons `coord_extremale_courbe` une classe dont les deux champs sont `coordonnée`, un réel, et `courbe`, un `ident_courbe`, et un `dico_cec` sera un dictionnaire augmenté de `coord_extremale_courbe`.

V.4.2. *L'algorithme de calcul de $\mathcal{P}$ tel que $\mathcal{A}(\mathcal{P})$ et $\mathcal{A}(\mathcal{B})$ soient équivalents*

La structure de l'algorithme est la suivante :

```
Arrangements_équivalents(arrangement 𝒜)
dico_paires_de_courbes LPC
dico_courbes ℒ
tableau_triangles 𝒯
begin
lecture_données(𝒜)
rectangles_superposés(𝒜,LPC)
construction_liste_triplets(LPC,𝒯)
vérification_intersections(𝒜,LPC)
liste_courbes_à_subdiviser(LPC,ℒ)
tant que ℒ ≠ ∅
     begin
     reconstruction_par_subdivision(LPC,ℒ)
     ar_subdivision(𝒜,ℒ)
     vérification_intersections(𝒜,LPC)
     liste_courbes_à_subdiviser(LPC,ℒ)
     end
équivalence_triangles(𝒜,𝒯)
écriture_résultats(𝒜)
end
```

Cet algorithme prend en entrée un `arrangement` $\mathcal{A}$, dont toutes les polycourbes sont dans leur écriture initiale, et donne à la sortie le même arrangement $\mathcal{A}$, mais dont les polycourbes ont subi des subdivisions de de Casteljau telles que dans leurs écritures courantes l'arrangement des polygones de contrôle respectifs soit équivalent à celui de polycourbes. Son idée est de chercher toutes les paires de courbes de Bézier ne faisant pas toutes les deux partie de la même polycourbe et dont nous ne pouvons pas affirmer que soit elles ne se coupent pas, soient elles vérifient (V.5), et de les introduire dans le dictionnaire `LPC` (c'est ce que font les procédures `rectangles_superposés` et `vérification_équivalence`). Ensuite, toutes les courbes de Bézier apparaissant dans au moins une des `paires_de_courbes` du dictionnaire `LPC` seront introduites dans le dictionnaire de courbes à subdiviser $\mathcal{L}$ (procédure `liste_courbes_à_subdiviser`). Si $\mathcal{L}$ n'est pas vide, nous subdivisons toutes les courbes de Bézier identifiées par $\mathcal{L}$ (procédure `ar_subdivision`) et nous recommençons. Si il l'est, la procédure `équivalence_triangles` prend en entrée le tableau $\mathcal{T}$ des triplets de courbes suscéptibles de déterminer (au moins) un triangle simple et assure, en subdivisant encore certaines des courbes de Bézier de la structure courante de l'arrangement si besoin est, l'équivalence de tous les triangles simples qui se correspondent dans $\mathcal{A}(\mathcal{B})$ et

$\mathcal{A}(\mathcal{P})$.

Les résultats présentés dans le paragraphe précédent de ce chapitre nous permettent d'affirmer, premièrement, que l'algorithme ci-dessus fournit réellement une écriture de la famille de polycourbes de Bézier donnée par $\mathcal{A}$ telle que l'arrangement des polygones de contrôle respectifs soit équivalent à l'arrangement de polycourbes, et deuxièmement, que cet algorithme converge.

Détaillons maintenant les procédures appelées par ce programme.

V.4.2.1. La procédure `rectangles_superposés`

Pour vérifier si deux polygones (dans notre cas convexes) ont une intersection vide, un premier critère efficace consiste à tester si les rectangles de côtés respectivement parallèles aux axes les englobant respectivement sont disjoints. Ainsi, pour trouver toutes les paires de courbes de Bézier ne faisant pas partie de la composition de la même polycourbe de l'arrangement donné et dont les B-polygones ne sont pas disjoints, nous commençons par chercher toutes les telles paires dont les B-polygones ont les rectangles englobants non disjoints. Le schéma de cette procédure est le suivant :

```
rectangles_superposés(arrangement A, dico_paires_de_courbes LPC)
dico_cec x_min, x_max
dico_paires_de_courbes temporaire
entiers i, j, k
begin
construction_liste_croissante_minima(A,x_min)
construction_liste_croissante_maxima(A,x_max)
i= 0; j= 0
tant que i < N
   begin
   tant que (j < N  et  x_min[j].coordonnée ≤ x_max[i].coordonnée)
      begin
      si x_min[j].curbe.quelle_polycourbe
                  ≠ x_max[i].curbe.quelle_polycourbe
         pour k allant de i à N − 1
            si (x_max[k].curbe,x_min[j].curbe) ∈ temporaire
               begin
               effacer (x_max[k].curbe,x_min[j].curbe) de temporaire
               si (min_y(x_min[j].curbe) ≤ max_y(x_max[k].curbe)
                                        et
                  min_y(x_max[k].curbe) ≤ max_y(x_min[j].curbe))
                  inserer (x_min[j].curbe,x_max[k].curbe) dans LPC
               end
            sinon
               inserer (x_min[j].curbe,x_max[k].curbe) dans temporaire
      j = j + 1
      end
   i = i + 1
```

```
26.         end
27.     end
```

Soient R_1 et R_2 deux rectangles (considérés des ensembles fermés) ayant les arêtes parallèles aux axes et supposons que les sommets de R_1 sont (a_1, b_1), (a_2, b_1), (a_2, b_2) et (a_1, b_2), avec $a_1 \leq a_2$ et $b_1 \leq b_2$, et les sommets de R_2 sont (α_1, β_1), (α_2, β_1), (α_2, β_2) et (α_1, β_2), avec $\alpha_1 \leq \alpha_2$ et $\beta_1 \leq \beta_2$. Il est très facile à voir qu'une condition nécessaire et suffisante pour avoir la relation $R_1 \cap R_2 \neq \emptyset$ est donnée par l'ensemble d'inégalités $\{a_1 \leq \alpha_2 \ ; \ \alpha_1 \leq a_2 \ ; \ b_1 \leq \beta_2 \ ; \ \beta_1 \leq b_2\}$, et c'est cette propriété qui a été utilisée dans l'algorithme ci-dessus.

La procédure commence par construire, lignes 6 et 7, deux dictionnaires de `coord_extrémale_courbe`, `x_min` et `x_max`. Un élément du dictionnaire `x_min` donne dans son champ `coordonnée` l'abscisse minimale du B-polygone de la courbe de Bézier désignée par son champ `courbe`. De la même manière, un élément du dictionnaire `x_max` donne dans son champ `coordonnée` l'abscisse maximale du B-polygone de la courbe de Bézier désignée par son champ `courbe`. L'ordre (total) considéré pour la classe `coord_extremale_courbe` est celui donné par son champ `coordonnée`. Chacun des dictionnaires `x_min` et `x_max` est de taille $N = \sum_{i=0}^{n} n_i$, où n est le nombre de polycourbes de l'arrangement et n_i est le nombre de courbes de Bézier composant la i-ème polycourbe. Le cycle des lignes 9-26 fait un parcours du dictionnaire `x_max`, et le cycle des lignes 11-24 fait un parcours en parallèle du dictionnaire `x_min`. Le dictionnaire `temporaire` est vide au début et nous y introduisons toute `paire_de_courbes (courbe1, courbe2)` telle que :

- `courbe1` et `courbe2` ne se situent pas sur la même polycourbe (le test de la ligne 13) ;
- la `paire_de_courbes (courbe2, courbe1)` n'appartient pas déjà à `temporaire`.

Le deuxième test de la ligne 11 nous assure que toute paire de courbes `(courbe1, courbe2)` candidate à être introduite dans `temporaire` satisfait la condition $a_1 \leq \alpha_2$, si nous considérons que les rectangles englobants des B-polygones des courbes `courbe1` et `courbe2` sont respectivement R_1 et R_2 et avec les notations ci-dessus. Si `(courbe2, courbe1)` est un élément de `temporaire`, nous déduisons que l'inégalité $\alpha_1 \leq a_2$ est également vérifiée ; dans ce cas, nous éliminons `(courbe2, courbe1)` de `temporaire` et nous testons si les deux rectangles ont une intersection non vide en vérifiant $b_1 \leq \beta_2$ et $\beta_1 \leq b_2$ (ligne 18). Si ces deux conditions sont satisfaites, `(courbe1, courbe2)` est introduite dans le dictionnaire `LPC` que la procédure donne comme résultat.

Cet algorithme fait appel à quatre sous procédures : `construction_liste_croissante-_minima`, `construction_liste_croissante_maxima`, `min_y` et `max_y`. Ces deux dernières calculent respectivement l'ordonnée minimale et maximale d'un polygone convexe donné (dans notre cas, B-polygone de la courbe de Bézier donnée). Grâce à la convexité du polygone à traiter, ces procédures sont de complexité $\mathcal{O}(\log d)$, d étant le nombre de sommets du polygone. Les deux autres procédures font appel respectivement à `min_x` et `max_x`, qui sont tout à fait similaires à `min_y` et `max_y` et retournent l'abscisse minimale et respectivement maximale d'un polygone convexe donné. En plus, `construction_liste_croissante_minima` et `construction_liste_croissante_maxima` construisent chacune un dictionnaire, l'insertion d'un nouvel élément se faisant en $\mathcal{O}(\log k)$, où k est la taille du dictionnaire déjà existent. Ainsi, `construction_liste_croissante_minima` et `construction_liste_croissante_maxima` seront de complexité $\mathcal{O}\big(N\log(N) + N\log(D)\big)$, où nous avons noté $D = \max_{i=0}^{n} \max_{j=0}^{n_i} m_{i,j}$ le degré maximal des courbes de Bézier intervennant dans l'arrangement et $N = \sum_{i=1}^{n} n_i$ le nombre total de courbes de Bézier intervenant dans l'arrangement.

Nous allons évaluer le coût de `rectangles_superposés` dans le cas le pire, qui survient quand chaque polycourbe est composée d'une unique courbe de Bézier et toutes deux courbes de Bézier ont les rectangles emboitants non disjoints. Dans ce cas, nous aurons la relation

$$\max_{j=0}^{N} \texttt{x\_min[j].coordonnée} \le \min_{i=0}^{N} \texttt{x\_max[i].coordonnée}$$

et pour `i` allant de 1 à $N-1$ l'algorithme n'exécutera que le premier test de la ligne 11, de complexité $\mathcal{O}(1)$, et le coût total sera donc $\mathcal{O}(N)$. Pour `i` $= 0$, les lignes 13-22 s'exécuteront pour chaque `j` entre 0 et $N-1$. À la j_0-ème itération, le dictionnaire `temporaire` aura au plus j_0N éléments et nous pouvons voir que la somme des coûts du test de la ligne 15 et de l'opération de la ligne 17 sera $\mathcal{O}(\log(j_0N))$. Les deux tests de la ligne 18 ont la complexité $\mathcal{O}(D)$. En sommant sur j, et comme les autres opérations du cycle des lignes 11-24 se font en temps constant à chaque itération, nous obtenons pour ce cycle la complexité $\mathcal{O}\big(N\log(N) + N\log(D)\big)$. Nous pouvons ainsi conclure que, dans le cas le pire, le coût total de la procédure `rectangles_superposés` est $\mathcal{O}\big(N\log(N) + N\log(D) + N\big)$.

V.4.2.2. La procédure `vérification_intersections`

Pour chaque paire de courbes identifiée par un élément du dictionnaire `LPC`, cette procédure vérifie si les conditions (V.1), (V.2) et (V.3) sont satisfaites, et si elles le sont, l'élément est éliminé de `LPC`. Son schéma est le suivant :

```
vérification_intersections(arrangement 𝒜, dico_paires_de_courbes LPC)
polygones 𝒫1, 𝒫2
entier cardinal
paire_de_courbes *ic1, *ic2
begin
*ic1 = premier élément de LPC
tant que ic1 ≠ NULL
   begin
   𝒫1 = 𝒜((*ic1).première_courbe)
   𝒫2 = 𝒜((*ic1).seconde_courbe)
   cardinal = intersection_lignes_polygonales(𝒫1, 𝒫2)
   si cardinal = 0
      effacer *ic1 du dictionnaire LPC
   si cardinal = 1
      si 𝒫1, 𝒫2 vérifient (V.2), (V.3)
         si 𝒫1, 𝒫2 vérifient (V.1)
            effacer *ic1 de la liste LPC
         sinon
            si vérification_avant/après_1(𝒜,LPC,ic1,ic2) = oui
               begin
               effacer *ic1 du dictionnaire LPC
               effacer *ic2 du dictionnaire LPC
               end
   si cardinal = −1
```

```
25.          si vérification_avant/après_2(𝒜,LPC,ic1,ic2) = oui
26.             begin
27.             effacer *ic1 du dictionnaire LPC
28.             effacer *ic2 du dictionnaire LPC
29.             end
30.       ic1 = (*ic1).suivant
31.       end
32.    end
```

Le cycle des lignes 7-31 s'effectue pour chaque élément du dictionnaire LPC et comme nous avons déjà mentionné, dans le cas le pire $M = \text{card}(\texttt{LPC}) = N(N-1)/2$. Les polygones $\mathcal{P}_1$ et $\mathcal{P}_2$ seront toujours des polygones simples et convexes et nous utilisons la notation $\mathcal{P}_i = \mathrm{P}_0^{(i)}\mathrm{P}_1^{(i)}\dots\mathrm{P}_{m_i}^{(i)}$, pour $i = 1, 2$.

Pour chaque paire de courbes correspondant à un élément de LPC, les polygones $\mathcal{P}_1$ et $\mathcal{P}_2$ sont respectivement les B-polygones des deux courbes de la paire. L'algorithme commence par appeler la procédure `intersection_lignes_polygonales`, qui, pour les deux polygones simples et convexes $\mathcal{P}_1$ et $\mathcal{P}_2$, rendra :

- 0, si

$$\left(\bigcup_{i=0}^{m_1-1}[\mathrm{P}_i^{(1)}\mathrm{P}_{i+1}^{(1)}]\right)\cap\left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)=\emptyset$$

et

$$\text{card}\Bigg(\left(\bigcup_{i=0}^{m_1-1}[\mathrm{P}_i^{(1)}\mathrm{P}_{i+1}^{(1)}]\right)\cap\text{Int}(\mathcal{P}_2)\Bigg)+\text{card}\Bigg(\left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)\cap\text{Int}(\mathcal{P}_1)\Bigg)=0$$

- -1, si

$$\left(\bigcup_{i=0}^{m_1-1}[\mathrm{P}_i^{(1)}\mathrm{P}_{i+1}^{(1)}]\right)\cap\left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)=\emptyset \tag{V.6}$$

et

$$\left(\bigcup_{i=0}^{m_1-1}[\mathrm{P}_i^{(1)}\mathrm{P}_{i+1}^{(1)}]\right)\cap\text{Int}(\mathcal{P}_2)\neq\emptyset \quad \text{ou} \quad \left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)\cap\text{Int}(\mathcal{P}_1)\neq\emptyset \tag{V.7}$$

- 1, si

$$\text{card}\Bigg(\left(\bigcup_{i=0}^{m_1-1}[\mathrm{P}_i^{(1)}\mathrm{P}_{i+1}^{(1)}]\right)\cap\left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)\Bigg)=1$$

- 2, si

$$\text{card}\Bigg(\left(\bigcup_{i=0}^{m_1-1}[\mathrm{P}_i^{(1)}\mathrm{P}_{i+1}^{(1)}]\right)\cap\left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)\Bigg)\geq 2$$

Ainsi, si le résultat de cette procédure est 0, nous sommes certains que ni les B-polygones, ni les courbes de Bézier correspondant à la `paire_de_courbes` traitée ne se coupent et donc celle-ci peut être éliminée de LPC (lignes 12-13). Si, au contraire, le résultat est 2, il est sûr que les deux courbes de Bézier ne satisfont pas les conditions requises, par conséquent la `paire_de_courbes` doit rester dans la liste LPC et donc l'algorithme se contente de passer à l'élément suivant de cette liste (ligne 30). Les cas nécessitant un traitement plus compliqué apparaîssent quand la procédure `intersection_lignes_polygonales` retourne 1 où -1.

Si le résultat est 1, il a un sens de nous poser la question si les B-polygones $\mathcal{P}_1$ et $\mathcal{P}_2$ satisfont les conditions (V.2) et (V.3) (ligne 15). Si ce n'est pas le cas, les deux courbes de Bézier devront être subdivisées et l'algorithme passe à l'itération suivante sans enlever l'élément courant de `LPC`. Si oui, alors nous vérifions si la condition (V.1) est satisfaite aussi, et si c'est le cas nous pouvons, en appliquant le lemme V.1, en déduire que cette paire de courbes ne soulève pas de problème pour l'équivalence des arrangements $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$ et donc nous l'effaçons de `LPC` (lignes 16-17). Si la condition (V.1) n'est pas vérifiée, la procédure `vérification_avant/après_1` est appelée.

Si nous nous situons dans le cas où cet appel est fait, il y a quatre positions relatives des B-polygones $\mathcal{P}_1$ et $\mathcal{P}_2$ possibles ; une de ces positions est présentée dans la figure suivante, les trois autres pouvant être représentées à partir de celle-ci en changeant les rôles des B-polygones $\mathcal{P}_1$ et $\mathcal{P}_2$ et en "symétrisant" la notation du B-polygone ayant une extrémité à l'intérieur de l'autre (pour le cas de la figure, $\mathrm{P}_0^{(1)}$ devient $\mathrm{P}_{m_1}^{(1)}$ et reciproquement). Étudions le fonctionnement de la procédure `vérification_avant/après_1` pour le cas présenté dans la figure, les autre cas seront traités de manière similaire.

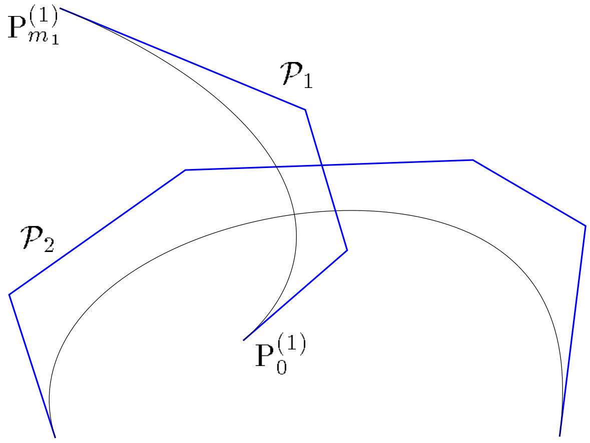


Figure V.16

On commence par tester si la courbe de Bézier correspondant au B-polygone $\mathcal{P}_1$ est la première de la polycourbe dont elle fait partie (autrement dit, nous vérifions si `(*ic1).première_courbe.quelle_courbe` $= 0$). Si c'est le cas, aucune autre opération n'est effectuée, et `*ic1` restera un élément de `LPC`, car la procédure retourne "`non`". Sinon, soit $\mathcal{P}_3$ le B-polygone de la courbe de Bézier précédant la courbe correspondant à $\mathcal{P}_1$ dans la composition de la polycourbe dont celle-ci fait partie. L'algorithme teste alors si $\widetilde{\mathcal{P}}_1 = \widetilde{\mathcal{P}}_3$. Au premier appel de la procédure `vérification_intersections` dans l'algorithme `Arrangements_équivalents`, toutes les polycourbes de l'arrangement sont dans leur écriture initiale et l'égalité ne peut évidemment pas être satisfaite, mais aux itérations suivantes elle devient possible. Si l'égalité n'est pas vérifiée, nous voudrons subdiviser les deux courbes désignées par `*ic1` et donc, pour ne pas exclure cet élément du dictionnaire `LPC`, `vérification_avant/après_1` retourne "`non`". Si oui, on vérifie les relations :

$$\operatorname{card}\left(\left(\bigcup_{i=0}^{m_3-1}[\mathrm{P}_i^{(3)}\mathrm{P}_{i+1}^{(3)}]\right)\cap[\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]\right)=1 \qquad \left(\bigcup_{i=0}^{m_3-1}[\mathrm{P}_i^{(3)}\mathrm{P}_{i+1}^{(3)}]\right)\cap\left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)=\emptyset$$

$$\left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)\cap[\mathrm{P}_0^{(3)}\mathrm{P}_{m_3}^{(3)}]=\emptyset \qquad \mathrm{card}\Big([\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]\cap[\mathrm{P}_0^{(3)}\mathrm{P}_{m_3}^{(3)}]\Big)=1$$

Le lemme V.2 nous autorise, si ces quatre conditions sont satisfaites, à effacer `*ic1` de la liste `LPC`. Plus, il existe un élément de `LPC[j]` désignant la paire de courbe qui correspond à $(\mathcal{P}_1, \mathcal{P}_3)$ (pas forcément dans cet ordre), le pointeur `ic2` indiquera cet élément et le même lemme nous dit que `*ic2` peut lui aussi être éliminé du dictionnaire de `paires_de_courbes`. Par conséquent, dans ce cas la procédure `vérification_avant/après_1` retournera "`oui`", mais elle déterminera d'abord le pointeur `ic2`. Ainsi, la procédure `vérification_intersections` executera les lignes 21 et 22, éliminant `*ic1` et `*ic2` du dictionnaire `LPC`. Si les quatres relations ci-dessus ne sont pas vérifiées, le résultat de la procédure sera "`non`" et `LPC` restera inchangée.

Si le résultat de la procédure `intersection_lignes_polygonales` est -1, nous nous retrouvons dans un cas qui est d'une certaine façon symétrique à celui déjà présenté. Nous savons qu'au moins une des deux relations de la condition (V.7) est satisfaite, supposons que c'est la première (l'autre cas se traite bien sûr de la même manière). La procédure `vérification_avant/après_2` commence par calculer le cardinal de l'ensemble $[\mathrm{P}_0^{(1)}\mathrm{P}_{m_1}^{(1)}]\cap[\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]$. Si ce cardinal est 0, nous nous trouvons dans une situation similaire à l'une des deux présentées figure V.17, et les deux courbes doivent être subdivisées, donc `vérification_avant/après_2` retourne "`non`".

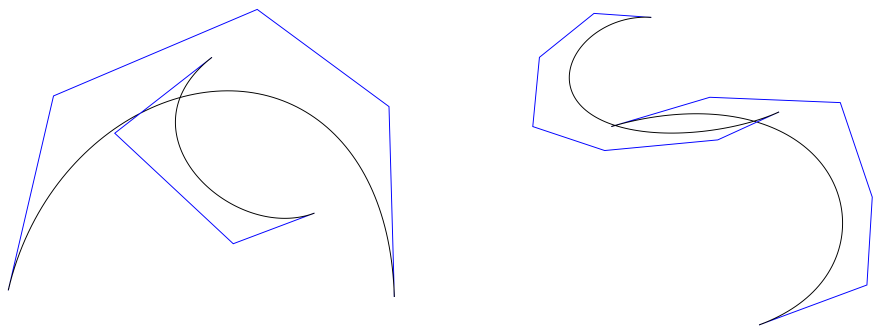
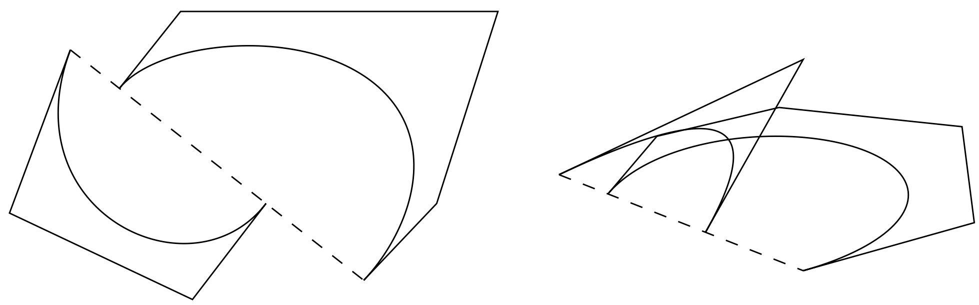


Figure V.17

Si le cardinal est $+\infty$, nous aurons la situation de la figure V.18(a), car celle de la figure V.18(b) ne satisfait pas (V.6). Dans ce cas aussi le résultat de `vérification_avant/après_2` sera "`non`".



(a) (b)

Figure V.18

Enfin, si le cardinal est 1, nous pouvons déduire que

$$\operatorname{card}\left(\left(\bigcup_{i=0}^{m_1-1}[\mathrm{P}_i^{(1)}\mathrm{P}_{i+1}^{(1)}]\right)\cap[\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]\right)=1$$

car $\mathcal{P}_1$ est convexe et la condition (V.6) est satisfaite. Alors un et seulement un des points $\mathrm{P}_0^{(1)}$ et $\mathrm{P}_{m_1}^{(1)}$ appartiendra à $\mathrm{Int}(\mathcal{P}_2)$, et supposons que ce point est $\mathrm{P}_{m_1}^{(1)}$.

Nous pouvons maintenant voir la symétrie mentionnée avec les cas traités par la procédure `vérification_avant/après_1` : le B-polygone $\mathcal{P}_1$ est assimilé au B-polygone $\mathcal{P}_3$ de la discussion précédente. La marche à suivre sera donc la suivante :

- nous vérifions si la courbe de Bézier correspondant à $\mathcal{P}_1$ est la dernière de la polycourbe dont elle fait partie (autrement dit, nous testons `(*ic1).première_courbe.quelle_courbe`$=\mathcal{A}($`(*ic1).première_courbe.quelle_polycourbe`$)$`.taille`-1)
- si elle l'est, `vérification_avant/après_2` retourne "`non`"
- si elle ne l'est pas, soit $\mathcal{P}_3$ le B-polygone correspondant à la courbe de Bézier succédant la courbe Bézier associée à $\mathcal{P}_1$ dans la composition de la polycourbe dont elle fait partie
- si $\widetilde{\mathcal{P}}_1 \neq \widetilde{\mathcal{P}}_3$, `vérification_avant/après_2` retourne "`non`"
- si $\widetilde{\mathcal{P}}_1 = \widetilde{\mathcal{P}}_3$, nous vérifions les relations

$$[\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]\cap[\mathrm{P}_0^{(3)}\mathrm{P}_{m_3}^{(3)}]=\emptyset \qquad \operatorname{card}\left(\left(\bigcup_{i=0}^{m_3-1}[\mathrm{P}_i^{(3)}\mathrm{P}_{i+1}^{(3)}]\right)\cap\left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)\right)=1$$

$$\operatorname{card}\left(\left(\bigcup_{i=0}^{m_2-1}[\mathrm{P}_i^{(2)}\mathrm{P}_{i+1}^{(2)}]\right)\cap[\mathrm{P}_0^{(3)}\mathrm{P}_{m_3}^{(3)}]\right)=1 \qquad \left(\bigcup_{i=0}^{m_3-1}[\mathrm{P}_i^{(3)}\mathrm{P}_{i+1}^{(3)}]\right)\cap[\mathrm{P}_0^{(2)}\mathrm{P}_{m_2}^{(2)}]=\emptyset$$

  (en fait, la deuxième et la troisième de ces quatre conditions impliquent les deux autres, dans les hypothèses existentes, et par conséquent suffisent)
- si l'une de ces relations n'est pas vraie, `vérification_avant/après_2` retourne "`non`"
- si toutes les quatre sont vérifiées, nous testons si $\mathcal{P}_2$ et $\mathcal{P}_3$ satisfont les conditions (V.2) et (V.3)
- si ils ne les satisfont pas, `vérification_avant/après_2` retourne "`non`"
- si oui, nous attribuons `ic2` tel que `*ic2` identifie la paire de courbes correspondant respectivement aux B-polygones $\mathcal{P}_2$ et $\mathcal{P}_3$, et `vérification_avant/après_2` retourne "`oui`"

Remarquons que le pointeur `ic2` calculé par l'une ou l'autre des procédures `vérification_avant/après_1` et `vérification_avant/après_2` indique toujours une `paire_de_courbes` supérieure à celle indiquée par le pointeur `*ic1` auquel il est associé, car si il est donné par `vérification_avant/après_1` et ne satisfait pas cette condition, alors il est facile à voir que `*ic1` aurait dû lui être associé par `vérification_avant/après_2` à l'itération lui correspondant dans le cycle `tant que` des lignes 7-30, ce qui est contradictoire, et une contradiction similaire est obtenue si `*ic2` est donné par `vérification_avant/après_2`.

Avant de faire l'étude de coût de cette procédure, un commentaire sur les algorithmes de `vérification_avant/après_1` et `vérification_avant/après_2` s'impose. Nous observons que ces procédures ne tirent aucun profit du fait que les objets à traiter ne sont pas de courbes de Bézier, mais des *polycourbes*, car quand $\widetilde{\mathcal{P}}_3$ est différent de $\widetilde{\mathcal{P}}_1$, pour `vérification_avant/après_1`, ou $\widetilde{\mathcal{P}}_2$, pour `vérification_avant/après_2`, les procédures retournent tout de suite "non". Nous sommes obligés de procéder ainsi à cause des configuration similaires à celle de la figure V.19, où $\mathcal{P}_1$ et $\mathcal{P}_2$ satisfont les condition nécessaires à l'appel de `vérification_avant/après_1` et $\mathcal{P}_3$ satisfait toutes les propriétés requises par cette procédure, sauf $\widetilde{\mathcal{P}}_3 = \widetilde{\mathcal{P}}_1$, et $\mathrm{card}\big(\mathcal{P}_2 \cap (\mathcal{P}_1 \cup \mathcal{P}_3)\big) \neq \mathrm{card}\big(\mathcal{B}_2 \cap (\mathcal{B}_1 \cup \mathcal{B}_3)\big)$.

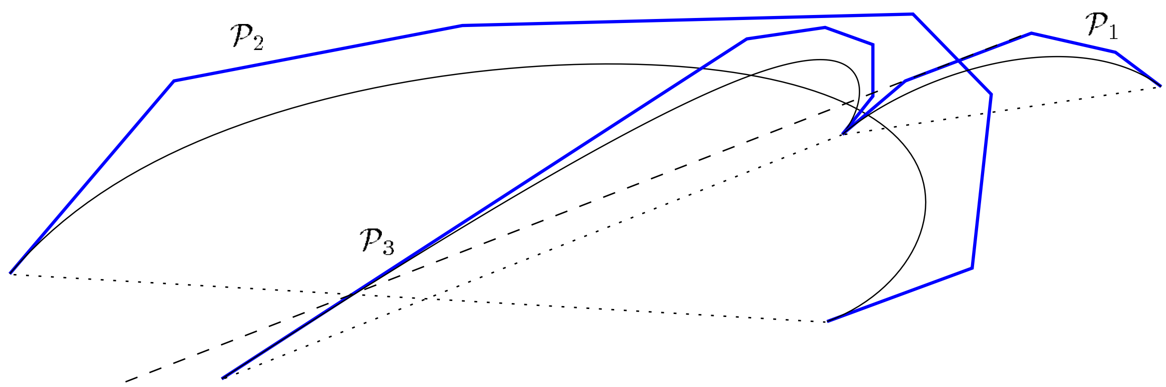


Figure V.19

Pour cette raison, nous devons imposer une condition supplémentaire à l'arrangement $\mathcal{A}(\mathcal{B})$ pour que notre algorithme converge : celle qu'aucune extrémité de *courbe de Bézier* (et non pas *polycourbe*) de l'écriture initiale de l'arrangement n'appartienne à une polycourbe autre que celle dont la courbe fait partie, hypothèse dont nous avons déjà vu la nécessité pour la preuve du lemme V.3.

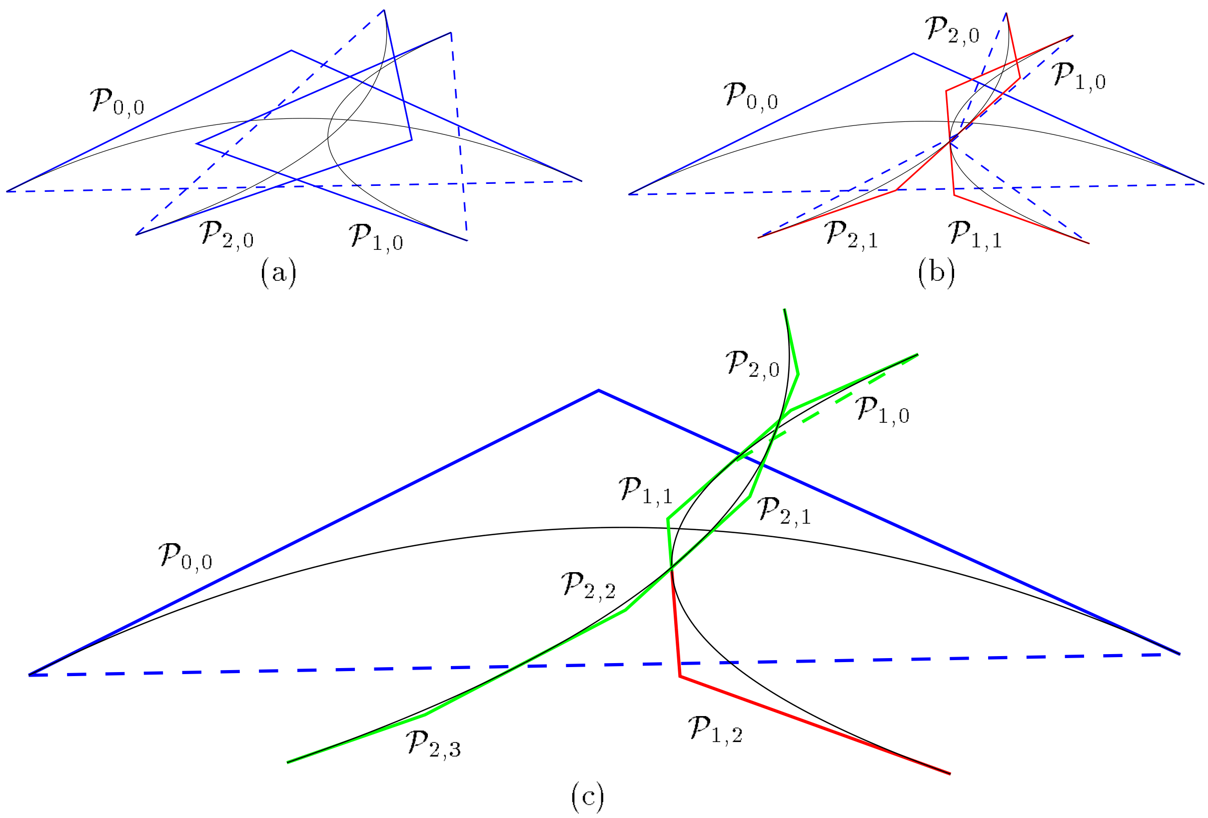


Figure V.20

D'un autre côté, nous n'utilisons pas "pleinement" le résultat V.3, car, dans les notations de ce lemme, nous vérifions les propriétés demandées uniquement pour $j_k = i_k \pm 1$, $\ell_k = i_k \pm$

1. Ainsi, pour l'arrangement de la figure V.20(a) (à la première itération) la `paire_courbes` $((1,0),(2,0))$ restera dans la liste de paires de courbes et $((0,0),(1,0))$ et $((0,0),(2,0))$ seront effacées ; après subdivision, dans la figure V.20(b) (la deuxième itération) les paires ne satisfaisant pas les conditions sont $((1,0),(2,0))$ et $((1,0),(2,1))$, tandis qu'après encore une subdivision le polygone $\mathcal{P}_{0,0}$ fera partie d'une paire de courbes ne satisfaisant pas les bonnes propriétés, car la valeur `cardinal` associée à la paire $((0,0),(1,0))$ sera 1, celle pour $((0,0),(1,1))$ sera -1 et celle pour $((0,0),(1,2))$ sera aussi -1 et c'est $\mathcal{P}_{1,2}$ qui vérifie les conditions de `vérification_avant/après_1` et non pas $\mathcal{P}_{1,1}$. Mais nous verrons dans la description de la procédure `reconstruction_par_subdivision` qu'en fait ce n'est que la première fois que le dictionnaire `LPC` est construit en vérifiant pour chaque paire de courbes n'appartenent pas à la même polycourbe de l'arrangement les conditions de `vérification_intersections`, aux itérations suivantes `LPC` se déduit à partir des paires de courbes qui ne satisfaisaient pas ces conditions à l'itération précédente.

Pour évaluer la complexité de `vérification_intersections`, nous devons étudier les coûts des procédures auxquelles cet algorithme fait appel.

La procédure `intersection_lignes_polygonales` est linéaire en la somme des tailles des deux polygones qu'elle traite, donc au pire linéaire en D, le maximum des degrés des courbes de Bézier de l'arrangement. Elle utilise l'algorithme classique de calcul de l'intersection de deux polygones convexes par décomposition en union de lignes polygonales d'abscisse croissante ; l'intersection de deux telles lignes polygonales se calcule en temps linéaire en leur nombre total de sommets, et la décomposition des B-polygones se fait en temps logarithmique, car elle consiste à trouver les sommets extrémaux sur une direction horizontale.

Le procédure `vérification_avant/après_1` est elle aussi linéaire en le maximum des degrés des trois polygones qui interviennent, qui peut être majoré par D, mais à ce coût s'ajoute celui de la recherche du pointeur `ic2` satisfaisant les conditions mentionnées, ce qui amène la complexité de cette procédure à $\mathcal{O}(\log(M))$. Il est facile à voir que la procédure `vérification_avant/après_2`, très semblable à la précédente, a la même complexité.

En sommant pour tous les éléments de `LPC`, nous obtenons pour `vérification_intersections` le coût $\mathcal{O}(M\log(M)+MD)$ (le test de la ligne 15 se fait en temps constant ; celui de la ligne 16 se fait en $\mathcal{O}(D)$; enfin, l'élimination des éléments de `LPC` se fait en $\mathcal{O}(\log(M))$). Dans le cas le pire, comme nous avons vu, $M = N(N-1)/2$ et ce coût sera $\mathcal{O}(N^2\log(N)+N^2D)$.

V.4.2.3. La procédure `liste_courbes_à_subdiviser`

Cette procédure construit, à partir du dictionnaire contenant les `paires_de_courbes` qui ne satisfont pas les critères donnés pour assurer l'équivalence des arrangements $\mathcal{A}(\mathcal{B})$ et $\mathcal{A}(\mathcal{P})$, le dictionnaire de courbes de Bézier participant à l'arrangement qui doivent être subdivisées. Son schéma est extrémement simple :

```
liste_courbes_à_subdiviser(dico_paires_de_courbes LPC, dico_courbes ℒ)
paire_de_courbes *ic = premier élément de LPC
begin
tant que ic ≠ NULL
   begin
   si (*ic).première_courbe ∉ ℒ
```

```
 7.          inserer (*ic).première_courbe dans ℒ
 8.       si (*ic).seconde_courbe ∉ ℒ
 9.          inserer (*ic).seconde_courbe dans ℒ
10.       si card ℒ = N
11.          sortir
12.       ic = (*ic).suivant
13.       end
14.    end
```

Le fonctionnement est aussi facile à suivre : pour chaque `paire_de_courbes` du dictionnaire `LPC`, on vérifie si les deux courbes sont déjà dans $\mathcal{L}$, et si elle ne le sont pas, on les y introduit.

$\mathcal{L}$ étant un dictionnaire, les tests des lignes 6 et 8 seront réalisés en $\mathcal{O}(\log k)$, où k est la taille courante du tableau, et les insertions des lignes 7 et 9 s'executent aussi en $\mathcal{O}(\log k)$.

Le test de la ligne 10 a été conçu dans l'espoir de réduire le nombre d'itérations à effectuer, qui dans le cas le pire est $N(N-1)/2$. Ainsi, si la taille de `LPC` est effectivement $N(N-1)/2$, ce test aménera le nombre d'itération, au pire, à $(N-1)(N-2)/2$, et au mieux, à $\lceil N/2 \rceil$. Dans le cas le pire, le coût de la procédure sera donc $\mathcal{O}(N^2 \log N)$, et dans le meilleur cas elle sera $\mathcal{O}(N \log N)$.

V.4.2.4. La procédure `ar_subdivision`

Cette procédure construit la nouvelle structure de l'arrangement, en subdivisant toutes les courbes de Bézier désignées par le dictionnaire $\mathcal{L}$. Elle utilise la procédure `subdivision` décrite dans le paragraphe III.3.2.3, mais, comme celle-ci traite des `polycourbes` et non pas des `tab_polycourbes`, une conversion de type doit être fait. Aussi, nous devons convertir les `ident_courbe` du dictionnaire $\mathcal{L}$ en pointeurs à des `courbes_de_Bézier` ; chacune de ces deux conversions se fait en temps linéaire en la taille de l'objet à convertir, elle coûteront donc $\mathcal{O}(N)$. En plus, après la subdivision de toutes le courbes de Bézier identifiées par $\mathcal{L}$ nous devons reconvertir chaque polycourbe, dans sa nouvelle écriture, de liste doublement chaînée au tableau, opération qui est toujours de complexité $\mathcal{O}(N)$. Le coût total de la procédure `ar_subdivision` sera, compte tenu de celui de `subdivision`, $\mathcal{O}(ND^2)$.

V.4.2.5. La procédure `reconstruction_par_subdivision`

Dans l'algorithme `Arrangements_équivalents`, nous utilisons cette procédure pour les itérations du cycle des lignes 12-18, plutôt que `rectangles_superposés`. La raison est que sa complexité est, comme nous le verrons, inférieure à celle de l'autre procédure, et tandis qu'au fur et à mesure des itérations le coût de `rectangles_superposés` serait croissant, car le nombre de courbes de Bézier intervenant dans l'arrangement croît, celui de `reconstruction_par_subdivision` décroît, car le cardinal de `LPC` est décroissant. Nous pouvons faire cette "substitution" de procédure, car il n'est pas difficile de voir que les propriétés requises pour les paires de courbes de Bézier de l'arrangement sont invariantes à la subdivisions (même si les conditions (V.4), (V.2) et (V.3) ne le sont pas à la subdivision non uniforme). Autrement dit, si une `paire_de_courbes` ne se trouve pas dans le dictionnaire `LPC` après son traitement par `vérification_intersections`, les deux courbes de Bézier que cette paire identifie ne se coupent pas ou ont exactement un point d'intersection, propriétés qui ne seront pas changées par subdivision. Par conséquent, à l'itération suivante nous aurons à chercher les "mauvaises" paires de courbes uniquement parmi celles provenant, par subdivision, des "mauvaises" paires de courbes à l'itération courante. Par ailleurs, cette façon de construire `LPC` à l'itération i à partir de `LPC` à l'itération $i-1$ fait aussi que l'exemple donnée dans la figure V.20 ne pose pas de problème, comme nous l'avons déjà mentionné. Le schéma de la procédure est le suivant :

```
1.   reconstruction_par_subdivision(dico_paires_de_courbes LPC,
                                                      dico_courbes ℒ)
2.   entiers i,a,b
3.   tableau_entiers ℐ
4.   tableau_paires_de_courbes copie = LPC
5.   begin
6.   effacement LPC
7.   création_liste_témoin(ℒ,ℐ)
8.   pout i allant de 0 à M − 1
9.      begin
10.     a = ℐ[copie[i].première_courbe]
11.     b = ℐ[copie[i].seconde_courbe]
12.     inserer transformée_1(copie[i],a,b) dans LPC
13.     inserer transformée_2(copie[i],a,b) dans LPC
14.     inserer transformée_3(copie[i],a,b) dans LPC
15.     inserer transformée_4(copie[i],a,b) dans LPC
16.     end
17.  end
```

Le tableau $\mathcal{I}$ est indexée sur tous les `indent_courbe` désignant une courbe de Bézier qui participe à l'arrangement. Si $i \in \{0, \ldots, n-1\}$ est l'indice d'une polycourbe dans l'arrangement et $j \in \{0, \ldots, n_i\}$ est l'indice d'une courbe de Bézier de la composition courante de $\mathcal{B}_i$, $\mathcal{I}[(i,j)]$ donnera le nombre de courbes de Bézier (toujours par rapport à la composition courante) de la polycourbe $\mathcal{B}_i$ qui correspondent à un indice dans cette polycourbe inférieur à j et qui apparaîssent dans la liste $\mathcal{L}$, donc qui seront subdivisées. Ainsi, après la subdivision de toutes les courbes de Bézier de $\mathcal{B}_i$ indiquées par le dictionnaire $\mathcal{L}$, la courbe qui était j-ème avant subdivision donnera les $j+\mathcal{I}[(i,j)]$-ème et $j+\mathcal{I}[(i,j)]+1$-ème courbes dans la nouvelle écriture de $\mathcal{B}_i$, si elle est subdivisée, et sera la $j+\mathcal{I}[(i,j)]$-ème courbe de $\mathcal{B}_i$, si elle ne l'est pas. La procédure `création_liste_témoin` crée la liste $\mathcal{I}$ en comptant pour chaque (i,j) les nombre de `indent_courbe` (i,k) avec $k < j$ qui sont des éléments de $\mathcal{L}$.

L'initialisation de la ligne 4 est évidemment un peu abusive, elle sous-entend une conversion de type. Les fonctions `transformée_i`, pour i allant de 1 à 4, donnent comme résultat, à partir de `copie[i]` (qui est un élément de `LPC` de l'itération précédente), les quatre paires de courbes obtenues en associant chacune des deux courbes de Bézier résultées de la subdivision de `copie[i].première_courbe` à chacune des deux courbes de Bézier résultées de la subdivision de `copie[i].seconde_courbe`. Les entiers `a` et `b` servent, comme présenté dans la description du tableau $\mathcal{I}$, au calcul des indices des nouvelles courbes de Bézier (obtenues par subdivision) dans leurs polycourbes respectives. Après l'exécution du cycle des lignes 8-16, le dictionnaire `LPC` aura 4 fois plus déléments.

Pour calculer la complexité de cette procédure, sommons les coûts de ses lignes. L'instruction de la ligne 4 s'execute en $\mathcal{O}(\text{card}(\texttt{LPC}))$, donc en $\mathcal{O}(M)$. La procédure **création_liste_témoin** coûte $\mathcal{O}(\left(\text{card}(\mathcal{L})\right))$, donc au pire $\mathcal{O}(N)$. Les instructions des lignes 10-11 sont chacune effectuées en temps constant et celles des lignes 12-15 en $\mathcal{O}(\log(k))$, k étant la taille actuelle de LPC. En conséquence le cycle des lignes 8-16 coûtera $\mathcal{O}(M\log(M))$. La complexité totale de la procédure sera ainsi $\mathcal{O}(N^2\log(N))$. Remarquons que la majoration de M par $N(N-1)$ est, surtout pour les dernières itérations, très grossière.

V.4.2.6. La procédure `construction_liste_triplets`

Cette procédure construit la liste de tous les triplets de courbes de Bézier susceptibles de représenter un triangle simple. Notre algorithme commence le traitement des triangles simples seulement après avoir fini les itérations assurant que toute paire de courbes de Bézier se coupe exactement autant de fois que la paire formée par leur B-polygones "étendus" respectifs et aussi que la paire de polygones support courante, c'est-à-dire que

$$\text{card}(\widetilde{\mathcal{B}}_{i_1,j_1}\cap\widetilde{\mathcal{B}}_{i_2,j_2})=\text{card}(\widetilde{\mathcal{P}}_{i_1,j_1}\cap\widetilde{\mathcal{P}}_{i_2,j_2})=\text{card}(\widetilde{\mathcal{S}}_{i_1,j_1}\cap\widetilde{\mathcal{S}}_{i_2,j_2})$$

pour tous $i_1\neq i_2\in\{1,\ldots,n\}$, $j_1\in\{0,\ldots,m_{i_1}\}$, $j_2\in\{0,\ldots,m_{i_2}\}$. Mais nous ne savons pas quelle est cette valeur commune des cardinaux des intersections, car l'arrangement initial ne satisfait généralement pas la condition $\text{card}(\mathcal{B}_{i_1,j_1}\cap\mathcal{B}_{i_2,j_2})\in\{0,1\}$ et, contrairement à la supposition faite pour prouver la correctitude et la convergence de l'algorithme, nous ne changeons pas la structure initiale de l'arrangement. Il n'est quand même pas difficile à voir que si pour un triplet $\left((i_1,j_1),(i_2,j_2),(i_3,j_3)\right)$ l'ordre des points $\left(\widetilde{\mathcal{P}}_{i_1,j_1}\cap\widetilde{\mathcal{P}}_{i_2,j_2}\right)\cup\left(\widetilde{\mathcal{P}}_{i_1,j_1}\cap\widetilde{\mathcal{P}}_{i_3,j_3}\right)$ sur $\widetilde{\mathcal{P}}_{i_1,j_1}$ est le même que l'ordre des points $\left(\widetilde{\mathcal{S}}_{i_1,j_1}\cap\widetilde{\mathcal{S}}_{i_2,j_2}\right)\cup\left(\widetilde{\mathcal{S}}_{i_1,j_1}\cap\widetilde{\mathcal{S}}_{i_3,j_3}\right)$ sur $\widetilde{\mathcal{S}}_{i_1,j_1}$, et pareil pour toutes les permutations circulaires des indices $1,2,3$, alors tous les pseudo-triangles définis par $\widetilde{\mathcal{B}}_{i_1,j_1}$, $\widetilde{\mathcal{B}}_{i_2,j_2}$ et $\widetilde{\mathcal{B}}_{i_3,j_3}$ seront équivalents aux ceux définis par $\widetilde{\mathcal{P}}_{i_1,j_1}$, $\widetilde{\mathcal{P}}_{i_2,j_2}$ et $\widetilde{\mathcal{P}}_{i_3,j_3}$, car nous n'aurons fait que traiter d'un seul coup tous les pseudo-triangles définis pas une triplet donné de courbes de l'arrangement initial à la place de les séparer d'abord et les traiter un par un ensuite.

Par conséquent, nous cherchons les triplets de courbes se coupant deux par deux dans l'écriture initiale de l'arrangement, car dans les conditions décrites (nous ne devons pas chercher que les triangles déterminés exactement par trois courbes de Bézier faisant respectivement partie de trois polycourbes distinctes, mais plus ; aussi, il y a bien plus de courbes de Bézier intervannant dans l'arrangement dans son écriture finale que dans l'initiale) il serait plus coûteux de faire autrement, et l'algorithme serait nettement plus compliqué. Dans cette situation, le tableau obtenu contiendra généralement un bon nombre de triplets designant trois courbes qui ne se coupent pas toutes deux, donc qui en fait ne nous interessent pas, mais la procédure appelée après, `équivalence_triangles`, se chargera de l'élimination de ces éléments de $\mathcal{T}$.

Ainsi, nous cherchons dans le dictionnaire LPC les indices `i, j, k` tels que l'ensemble de `ident_courbe`

$$\Big\{\texttt{LPC[i].première\_courbe, LPC[i].seconde\_courbe, LPC[j].première\_courbe,}$$
$$\texttt{LPC[j].seconde\_courbe, LPC[k].première\_courbe, LPC[k].seconde\_courbe}\Big\}$$

ait exactement trois éléments et que dans cette écriture chacun apparaisse exactement deux fois. Pour chaque tel triplet d'indices, nous ordonnons (par ordre lexicographique) les trois `ident_courbe` distincts de l'ensemble ci-dessus et introduisons le `triangle` ainsi obtenu dans le tableau $\mathcal{T}$ (l'ajout d'un nouvel `triangle` se fait toujours à la fin du tableau

; le fait que LPC est un dictionnaire, donc fournit les paires_de_courbes en ordre, nous garantit le bon ordre des eléments de $\mathcal{T}$). Voici le schéma de la procédure :

```
1.  construction_liste_triplets(dico_paires_de_courbes LPC,
                                                  tableau_triplets 𝒯)
2.  paire_de_courbes *i,*j
3.  triangle T
4.  begin
5.  *i = premier élément de LPC
6.  tant que i ≠ NULL
7.     begin
8.     j = (*i).suivant
9.     tant que (j ≠ NULL  et  (*j).première_courbe=(*i).première_courbe)
10.       begin
11.       si ((*i).seconde_courbe,(*j).seconde_courbe) ∈ LPC
12.          begin
13.          T.première_courbe = (*i).première_courbe
14.          T.seconde_courbe = (*i).seconde_courbe
15.          T.troisième_courbe = (*j).seconde_courbe
16.          ajouter T à 𝒯
17.          end
18.       j = (*j).suivant
19.       end
20.    i = (*i).suivant
21.    end
22. end
```

Le test de la ligne 11 s'execute $\mathcal{O}(N^3)$ fois quand card(LPC) $= N(N-1)/2$ et il s'execute en $\mathcal{O}(\log(N))$; comme toutes les autres instructions de la procédure sont de complexité $\mathcal{O}(1)$, le coût de construction_liste_triplets est $\mathcal{O}(N^3 \log(N))$.

Remarquons qu'une procédure construisant la liste de tous les triplets de courbes de Bézier de la structure initiale de l'arrangement qui sont susceptibles de former un triangle simple en testant tout simplement les intersections de tous les triplets possibles de B-polygones est de complexité $\mathcal{O}(N^3 D)$, qui généralement est inférieure à la complexité de notre procédure (N est suffisament grand par rapport à D).

V.4.2.7. La procédure équivalence_triangles

Comme précisé au début du paragraphe V.4.2.6, pour tester l'équivalence des trianlges simples trouvés par la procédure précédente nous avons choisi une méthode consistant à comparer les ordres respectifs dans lesquels $\widetilde{\mathcal{P}}_{i_1,j_1}$ coupe $\widetilde{\mathcal{P}}_{i_2,j_2}$ et $\widetilde{\mathcal{P}}_{i_3,j_3}$ et $\widetilde{\mathcal{S}}_{i_1,j_1}$ coupe $\widetilde{\mathcal{S}}_{i_2,j_2}$ et $\widetilde{\mathcal{S}}_{i_3,j_3}$ (et de même pour toutes les permutations des indices).

Avant de décrire cette procédure, introduisons une nouvelle structure de données, le segm_inters, ayant deux champs, segment et avec_qui. Si nous avons trois polygones, $\mathcal{P}_1$, $\mathcal{P}_2$ et $\mathcal{P}_3$, un segm_inters, notons-le si, attaché à, par exemple, $\mathcal{P}_1$, indiquera une intersection de ce polygone avec l'un des deux autres : nous aurons

$$[\mathrm{P}^{(1)}_{\mathrm{si.segment}}\mathrm{P}^{(1)}_{\mathrm{si.segment+1}}] \cap \mathcal{P}_{\mathrm{si.avec\_qui}} \neq \emptyset$$

```
 1. équivalence_triangles(arrangement 𝒜, tableau_triplets 𝒯)
 2. triangle T
 3. liste_segm_inters SI
 4. polygones contr1, contr2, contr3, supp1, supp2, supp3
 5. liste_segm_inters c1, c2, c3, s1, s2, s3
 6. begin
 7. pour tout T ∈ 𝒯
 8.   begin
 9.   création_poly_contr/supp(𝒜,T.première_courbe,contr1,supp1)
10.   création_poly_contr/supp(𝒜,T.seconde_courbe,contr2,supp2)
11.   création_poly_contr/supp(𝒜,T.troisième_courbe,contr3,supp3)
12.   si supp1 ∩ supp2 ≠ ∅   et   supp1 ∩ supp3 ≠ ∅   et   supp2 ∩ supp3 ≠ ∅
13.     begin
14.     liste_des_intersections(contr1,contr2,contr3,
                 supp1,supp2,supp3,c1,c2,c3,s1,s2,s3)
15.     tant que comparaison(c1,s1) = faux   ou
              comparaison(c2,s2) = faux   ou   comparaison(c3,s3) = faux
16.       begin
17.       subdiviser_alternances(𝒜,T.première_courbe,s1)
18.       subdiviser_alternances(𝒜,T.seconde_courbe,s2)
19.       subdiviser_alternances(𝒜,T.troisième_courbe,s3)
20.       création_poly_contr/supp(𝒜,T.première_courbe,contr1,supp1)
21.       création_poly_contr/supp(𝒜,T.seconde_courbe,contr2,supp2)
22.       création_poly_contr/supp(𝒜,T.troisième_courbe,contr3,supp3)
23.       liste_des_intersections(contr1,contr2,contr3,
                          supp1, supp2,supp3,c1,c2,c3,s1,s2,s3)
24.       end
25.     end
26.   end
27. end
```

La procédure `création_poly_contr/supp` construit, pour un `ident_courbe ic` donné, le polygone union des B-polygones de toutes les courbes de Bézier qui proviennent, dans l'écriture courante de l'arrangement, de la courbe de Bézier de la structure initiale de l'arrangement designée par `ic`, et similairement pour les polygones support (l'identification des courbes obtenues par la subdivision d'une courbe donnée est possible grâce au champ `rang` d'une `courbe_de_Bézier`). Il est évident que le polygone `supp1` aura moins de sommets que `contr1`, et de même pour `supp2` et `contr2` et respectivement `supp3` et `contr3`. Mais aussi, nous aurons card($\mathtt{supp1} \cap \mathtt{supp2}$) = card($\mathtt{contr1} \cap \mathtt{contr2}$), card($\mathtt{supp1} \cap \mathtt{supp3}$) = card($\mathtt{contr1} \cap \mathtt{contr3}$) et card($\mathtt{supp2} \cap \mathtt{supp3}$) = card($\mathtt{contr2} \cap \mathtt{contr3}$), car l'arrangement a subi le traitement du cycle des lignes 12-18 (de l'algorithme `Arrangements_équivalents`). Ainsi, quand nous voulons vérifier si les courbes de Bézier "incriminées" (designées par le `triangle T` dans la structure initiale de $\mathcal{A}$) ont vraiment une intersection non vide deux par deux, nous regardons si cette propriété est satisfaite par `supp1`, `supp2` et `supp3`, c'est moins coûteux (ligne 12). Si ce n'est pas le cas, les trois courbes ne forment aucun pseudo-triangle et nous passons à l'élément suivant de $\mathcal{T}$.

Si les trois courbes de Bézier se coupent deux par deux, nous associons à chacune d'entre elles deux listes de segm_inters, créant ainsi les liste_segm_inters c1, c2, c3, s1, s2, s3. c1 et s1 correspondent à $\mathcal{A}_{\mathrm{init}}$(T.première_courbe) : c1 indique, dans l'ordre, toutes les intersections de contr1 avec contr2 et contr3, et s1 de même, pour supp1, supp2 et supp3. De la même façon, les listes c2, s2, respectivement c3, s3 ont associées à $\mathcal{A}_{\mathrm{init}}$(T.seconde_courbe), respectivement $\mathcal{A}_{\mathrm{init}}$(T.troisième_courbe). Nous avons card(c1) = card(contr1 $\cap$ contr2) + card(contr1 $\cap$ contr3) et une relation analogue est valable pour card(s1), d'où nous déduisons tout de suite card(c1) = card(s1), card(c2) = card(s2) et card(c3) = card(s3). La procédure comparaison vérifie, pour deux liste_segm_inters de la même longueur, si les deux listes d'entiers respectivement obtenues en considérant uniquement le champ avec_qui de chaque élément (de chacune des listes_segm_inters) sont identiques, et retourne "vrai" si c'est le cas et "faux" sinon.

Il n'est pas difficile à voir que si le test de la ligne 15 rend "vrai", tous les pseudo-triangles éventuellement déterminés par les trois courbes couramment traitées satisfont les "bonnes" propriétés, et nous passons alors à l'élément suivant de la liste $\mathcal{T}$. Si ce test rend "faux", il faudra subdiviser certaines (ou toutes) des courbes de Bézier concernées ; subdiviser toutes ces courbes peut ne pas être nécessaire. Un test déterminant exactement quelles courbes il faut subdiviser serait bien trop complexe, ce pourquoi nous avons choisi un critère qui a l'avantage d'être très simple, tout en ayant l'inconvenient de nous faire subdiviser plus des courbes de Bézier qu'il n'en est nécessaire. Nous subdiviserons donc la i-ème courbe correspondant à $\mathcal{A}_{\mathrm{init}}$(T.première_courbe) si et seulement si s1[i].avec_qui $\neq$ s1[i+1].avec_qui ou s1[i].avec_qui $\neq$ s1[i-1].avec_qui (au cas où l'un des éléments s1[i-1] et s1[i+1] n'existe pas, nous considérons uniquement celui qui existe). Le même critère s'applique bien sûr aux courbes provenant de $\mathcal{A}_{\mathrm{init}}$(T.seconde_courbe) et $\mathcal{A}_{\mathrm{init}}$(T.troisième_courbe). Après ces subdivisions (lignes 17-19), le proccessus est repris jusqu'à ce que le test de la ligne 15 rende "vrai", moment où nous passons à l'élément suivant de $\mathcal{T}$.

À la différence des autre procédures détaillées ici, nous ne pouvons pas calculer le coût de équivalence triangles. Ceci est dû au fait, premièrement, que nous ne savons pas (et ne pouvons pas évaluer autrement que de manière experimentale) le nombre d'itérations qui seront nécessaires dans le cycle des lignes 15-24. Deuxièmement, pour donner la complexité de la procédure création_poly_contr/supp nous devrions savoir combien de courbes ont été obtenues, au cours des itérations du cycle des lignes 12-18 de Arrangements_équivalents, à partir d'une courbe de Bézier donnée de l'arrangement initial. Pour ça, nous aurions besoin d'une évaluation du nombre de telles itérations effectuées, qui non plus ne peut être faite théoriquement.

Supposons que le cycle des lignes 12-18 de Arrangements_équivalents requiert κ_1 itérations et majorons par κ_2 le nombre d'itérations nécessaires dans le cycle des lignes 15-24 d'équivalence_triangles. Il y aura alors au plus 2^{κ_1} courbes provenant d'une même courbe de Bézier de l'écriture initiale de l'arrangement et par conséquent création_poly_contr/supp sera de complexité $\mathcal{O}(2^{\kappa_1}D)$, D étant toujours le degré maximal des courbes de Bézier de $\mathcal{A}$. Le test de la ligne 12 est linéaire en la somme des cardinaux des polygones supp1, supp2, supp3, et coûte ainsi $\mathcal{O}(2^{\kappa_1})$. La procédure liste_des_intersections est elle aussi linéaire en le maximum des sommes des cardinaux des polygones contr1, contr2, contr3, donc de complexité $\mathcal{O}(2^{\kappa_1}D)$. La procédure comparaison est linéaire en le cardinal des listes qu'elle traite. Comme deux courbes polynômiales distinctes peuvent avoir en commun au plus leur degré maximal au carré points (n'oublions pas que l'arrangement $\mathcal{A}$ est simple, les courbes de Bézier ne peuvent pas être confondues sur intervalles), le test de la ligne 15 coûte $\mathcal{O}(D^2)$. Pour les subdivisions (lignes 17-19), le cas le pire consiste à subdiviser toutes les courbes de Bézier impliquées, doublant ainsi leur nombre ; cette opération est de complexité $\mathcal{O}(2^{\kappa_1+i}D^2)$, quand nous sommes à l'$i$-ème itération dans le cycle des lignes 15-24. Ensuite, à leurs appels à l'intérieur de ce cycle, les procédures `création_poly_contr/supp` et `liste_des_intersections` auront toutes les deux le coût $\mathcal{O}(2^{\kappa_1+i}D)$ à la i-ème itération. En sommant, nous obtenons pour le cycle la complexité $\mathcal{O}(2^{\kappa_1+\kappa_2}(D^2+D)+\kappa_2 D^2)$. Par conséquent, la complexité de la procédure `équivalence_triangles` sera

$$\mathcal{O}\big((2^{\kappa_1+\kappa_2}(D^2+D)+2^{\kappa_1}D+\kappa_2 D^2)\,\mathrm{card}(\mathcal{T})\big)$$

La liste $\mathcal{T}$ a, au cas le pire, $\mathrm{C}_N^3 = N(N-1)(N-2)/6$ éléments.

V.4.3. *La complexité de l'algorithme* `Arrangements_équivalents`

De la même façon que pour l'algorithme de calcul de l'enveloppe convexe topologique d'une polycourbe fermée, le coût de l'algorithme ne peut pas être évalué en fonction uniquement de la taille des données, car le nombre d'itérations nécessaires pour que les conditions d'équivalence soient satisfaites par $\mathcal{A}$ reflète les propriétés "qualitatives" et non pas "quantitatives" de l'arrangement. Pour donner quand même une idée de la complexité de l'algorithme, supposons, comme dans le paragraphe précédent, que les instructions du cycle des lignes 12-18 seront exécutées κ_1 fois. Nous verrons dans la section suivante que de toute manière le coût théorique que nous obtiendrons ici est largement supérieur au coût moyen obtenu expérimentalement, et par conséquent nous allons négliger dans les expressions des coûts des procédures qu'`Arrangements_équivalents` appelle tous les termes à part les plus significatifs.

Soit N_0 le nombre de courbes de Bézier intervennant dans l'écriture initiale de l'arrangement. La procédure `rectangles_superposés` sera alors de complexité $\mathcal{O}(N_0\log N_0 + N_0\log D)$. À l'appel de la ligne 9, `vérification_intersections` coûtera $\mathcal{O}(N_0^2\log N_0 + N_0^2 D)$. À la ligne 10, `liste_courbes_à_subdiviser` est de complexité $\mathcal{O}(N_0^2\log N_0)$. Pour voir quels sont les coûts des appels faits à l'intérieur du cycle des lignes 12-18, il nous faut évaluer la taille (en courbes de Bézier) N_i de l'arrangement à l'i-ème itération. Au cas le pire, à chaque itération toutes les courbes de Bézier de l'arrangement sont subdivisées, donc à chaque itération N_i double. Nous aurons ainsi $N_i = 2^{i-1}N_0$. En conséquence, à l'i-ème itération les lignes 14-17 auront un coût total de $\mathcal{O}(2(i-1)2^{2(i-1)}N_0^2\log N_0 + 2^{2(i-1)}N_0^2D^2)$ et nous obtenons, en sommant, la complexité du cycle des lignes 12-18 $\mathcal{O}(\kappa_1 2^{2\kappa_1}N_0^2\log N_0 + 2^{2\kappa_1}N_0^2D^2)$. En y ajoutant les coûts des procédures qui ne sont pas comprises dans le cycle, nous obtenons le coût de l'algorithme `Arrangements_équivalents` : $\mathcal{O}(\kappa_1 2^{2\kappa_1}N_0^2\log N_0 + 2^{2\kappa_1}N_0^2D^2 + 2^{\kappa_1+\kappa_2}N_0^3D^2 + N_0^3\log N_0)$.

Il faut remarquer que le "cas le pire" est assez dificile à avoir, voir impossible dans les dernières itérations (aussi bien dans le cycle des lignes 12-18 d'`Arrangements_équivalents` que dans le cycle des lignes 15-24 d'`équivalence_triangles` et en ce qui concérne le cardinal de la liste $\mathcal{T}$ dans cette dernière procédure), à moins d'avoir un très faible nombre de courbes de Bézier qui interviennent, ou alors des valeurs de κ_1 et κ_2 très faibles. En plus, même en considérant le cas le pire pour la structure initiale de l'arrangement, où chaque courbe de Bézier représente une polycourbe, il est évident que si au début nous avons effectivement N_0 polycourbes à une courbe de Bézier chacune, après une itération, donc la subdivision de partie ou toutes les courbes de Bézier, ceci n'est plus possible.

V.5. Résultats numériques

Le programme issu de l'implémentation des algorithmes présentés dans la section précédente a été testé sur un échantillon de 60 arrangements. Chacun des arrangements contient autour de 100 courbes de Bézier et nous avons fait varier le nombre de courbes composant chaque polycourbe de l'arrangement entre 1 et 10. Nous avons ainsi 10 arrangements de 100 polycourbes composées chacune d'une seule courbe de Bézier, 10 arrangements de 20 polycourbes composés chacune de 5 courbes de Bézier et 10 arrangements de 10 polycourbes composées chacune de 10 courbes de Bézier, le reste des exemples étant uniformement distribués entre les arrangements de polycourbes à 2, 3, 4, 6, ..., 9 courbes de Bézier.

Le tableau suivant donne quelques valeurs correspondant aux arrangements initiaux, ceux qui n'ont pas encore été traités par le programme. Ces valeurs sont n, le nombre moyen de polycourbes par arrangement, N_0, le nombre moyen de courbes de Bézier par arrangement, et D, le degré moyen de ces courbes de Bézier.

	n	N_0	D
Arrangement initial	$33,52$	$100,2$	$3,51$

Pour les arrangements résultant après traitement, ceux pour lesquels l'arrangement des polygones de contrôle est équivalent à l'arrangement de polycourbe (et à l'arrangement des polygones support, si la situation mentionnée à la fin de la section V.3 n'apparaît pas), nous avons trouvé que les valeurs suivantes donnent une bonne idée du fonctionnement de l'algorithme et de sa complexité :

- N, le nombre de courbes de Bézier (de la structure finale de l'arrangement) ;
- κ_1, le nombre d'itération nécessaires dans le cycle des lignes 12-18 de l'algorithme `Arrangements_équivalents` ;
- $\tau = \mathrm{card}(\mathcal{T})$, le nombre de triplets que la liste $\mathcal{T}$ contient après l'appel de la procédure `construction_liste_triplets` (ligne 9) ;
- τ_{eff}, le nombre de triplets de $\mathcal{T}$ qui forment effectivement un triangle simple ;
- τ_{non-eq}, le nombre de triangles simples ne satisfaisant pas les conditions d'équivalence et devant être subdivisés ;
- κ_2, le nombre d'itérations nécessaires pour que les triangles simples qui ne satisfaisaient les conditions d'équivalence le fassent.

Le tableau qui suit donne pour tous ces paramètres les valeurs moyennes et les valeurs maximales pour l'échantillon étudié.

Si, pour un arrangement donné, le nombre d'opérations à effectuer pour que les conditions d'équivalence soient satisfaites est celui donné par la complexité théorique de la sous section V.3.3, la forme finale de l'arrangement devrait contenir $2^{\kappa_1+\kappa_2}N_0$ courbes de Bézier. Donc dans notre cas, la valeur moyenne de N devrait être 12800, d'un ordre de grandeur supérieure à celle effectivement obtenue (et encore quatre fois plus grande que la pire valeur de nos exemples).

	N	κ_1	τ	τ_{eff}	τ_{non-eq}	κ_2
Valeur moyenne	$1202,38$	$6,69$	$17373,6$	$5287,7$	$77,15$	$1,285$
Valeur maximale	3296	10	81486	47913	278	10

Aussi, le nombre le pire (le plus grand) de triplets de l'ensemble $\mathcal{T}$ est $N_0(N_0-m)(N_0-2m)$, où m est le nombre de courbes de Bézier constituant une polycourbe, considéré ici le même pour toutes les polycourbes de l'arrangement. Pour notre échantillon, il résulte $\tau = 720000$ même en mettant $m = 10$, le meilleur cas, ce qui est 40 fois la valeur que nous avons obtenue en moyenne et toujours 9 fois plus pour le pire exemple. Il faut en plus remarquer qu'en moyenne deux tiers de ces triplets ne constituent pas des triangles simples et sont éliminés tout de suite et à un coût réduit, car nous testons l'intersections des polygones support correspondant aux courbes de Bézier identifiées par un tel triplet ; ces polygones ont au cas le pire 1024 sommets et en moyenne pas plus de 12 sommets. Et encore, parmi ceux-là seulement 1/70 ne satisfont pas les conditions d'équivalence des triangles simples et nous devrons subdiviser les trois courbes de Bézier qu'ils identifient. Dans la plupart des cas, une seule subdivision de ces courbes suffira.

Nous avons calculé aussi le nombre maximal de subdivisions qui ont été appliquées aux courbes de l'échantillon d'arrangements, et il est égal à 16 (d'où, évidemment, la conclusion que le pire cas pour κ_1 et le pire cas pour κ_2 ne sont pas survenus pour le même exemple). La figure suivante présente la repartition des courbes de Bézier par nombre de subdivisions appliquées (voir la définition III.10 pour la notion de "courbe de Bézier subdivisée k fois"), en pourcentage sur le nombre total de courbes de l'échantillon.

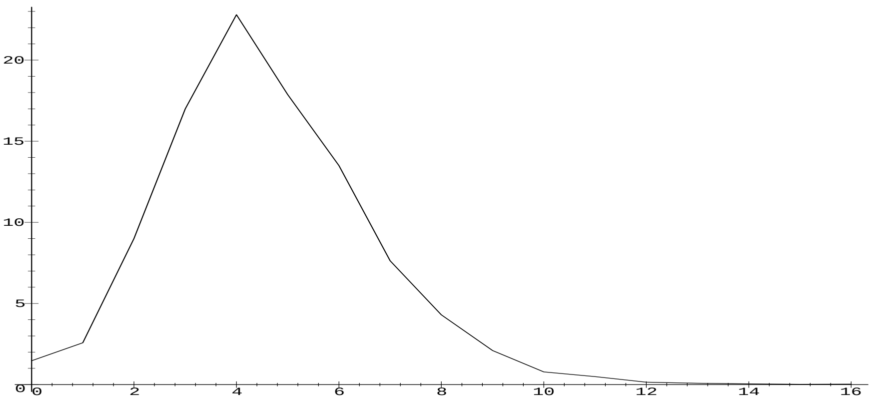


Figure V.21

Ce graphe a l'allure d'une gaussienne, comme c'était à attendre, car à la différence du cas de l'enveloppe convexe les subdivisions se font "dans toute la masse" de l'arrangement. Le pic correspond à 4 subdivisions et est de 22,81% ; les valeurs correspondant à 13 ou plus subdivisions sont inférieures à 0,1%.

Précisons aussi que le nombre moyen de sommets d'un arrangement dans son écriture finale est 4208,59 et nous voyons ainsi quelle est l'importance du corollaire V.7, car nous avons aussi un arrangement, celui des polygones support, ayant en moyenne 1232,38 sommets (soit plus de trois fois moins que celui de polygones de contrôle) et qui est équivalent à l'arrangement de polycourbes.

Nous avons précisé dans les discussions de complexité des diverses procédures appelées par l'algorithme que le cas le pire survient quand l'arrangement initial est composé de N_0 poycourbes à une courbe de Bézier chacune (sauf pour la procédure de subdivision). Le tableau suivant illustre cette affirmation, en présentant les mêmes valeurs que le précédent respectivement pour des arrangements de 100 polycourbes à 1 courbe de Bézier, 20 polycourbes à 5 courbes de Bézier et 10 polycourbes à 10 courbes de Bézier.

Dans ce tableau, les valeurs de κ_1 et κ_2 sont sensiblement approchées (et approchées de la valeur du tableau précédent, correspondant à la totalité de l'échantillon). Nous déduisons que les valeurs très différentes des tailles (en courbes de Bézier) des arrangements finaux pour les trois cas résultent du nombre de courbes à subdiviser à chaque itération, qui est bien plus important pour les polycourbes à une seule courbe de Bézier que pour celle à dix courbes de Bézier. La même observation est valable en ce qui concerne le traitement des triangles simples, ce qui est d'ailleurs reflété par les valeurs de τ, τ_{eff} et τ_{non-eq} dans les trois cas.

		N	κ_1	τ	τ_{eff}	τ_{non-eq}	κ_2
100x1	Valeur moyenne	$2474,6$	$7,7$	$52330,2$	$16709,6$	$179,8$	$1,31$
	Valeur maximale	3296	9	81486	47913	278	10
20x5	Valeur moyenne	$1102,7$	$6,6$	$10488,9$	$2620,5$	$63,8$	$1,3$
	Valeur maximale	1729	8	15749	3824	124	6
10x10	Valeur moyenne	$858,44$	6	$4790,56$	$1361,56$	$44,9$	$1,31$
	Valeur maximale	1184	7	12197	5401	103	6

La figure suivante présente la distribution des courbes par nombre de subdivisions appliquées respectivement pour les polycourbes à 10 courbes, à 5 courbes et à une courbe de Bézier. Si pour les exemples d'arrangements de polycourbes à une courbe de Bézier le nombre maximal de subdivisions appliquées est 16 (d'ailleurs ces exemples fournissent, dans l'échantillon intégral, la totalité des courbes de Bézier subdivisées 16 fois), pour les exemples d'arrangements de polycourbes à 5 courbes de Bézier cette valeur descend à 12 et pour ceux à 10 courbes à 10.

Ces graphes aussi ressemblent à des gaussiennes. Le pic pour le premier (correspondant aux arrangements de dix polycourbes à dix courbes) correspond à 4 subdivisions et est de 29,95% (avec 24% pour 3 subdivisions) ; le pic pour le deuxième (correspondant aux arrangements de vingt polycourbes à cinq courbes) est atteint pour 4 subdivisions et représente 24,35% (avec 22,46% pour cinq subdivisions) ; enfin, le pic du troisième graphe est de 21,57% et est atteint pour 6 subdivisions (avec 19% pour 7 subdivisions).

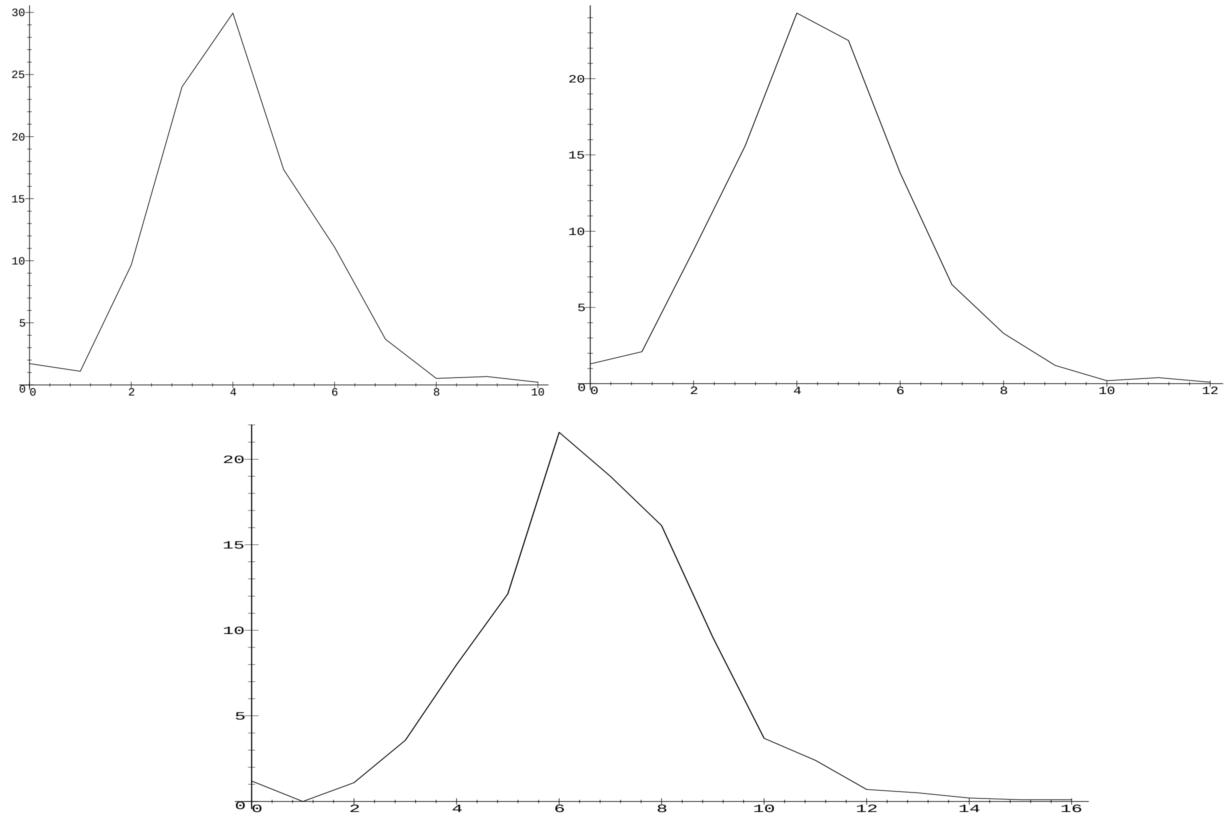


Figure V.22

V.6. Exemples

Nous n'allons pas donner beaucoup d'exemples faisant partie de notre échantillon, car le nombre élevé de courbes, respectivement polygones, fait les faces des arrangements difficiles à distinguer. Les figures sont groupées par trois, **A** est l'arrangement de polycourbes, **B** celui de polygones de contrôle et **C** l'arrangement de polygones support.

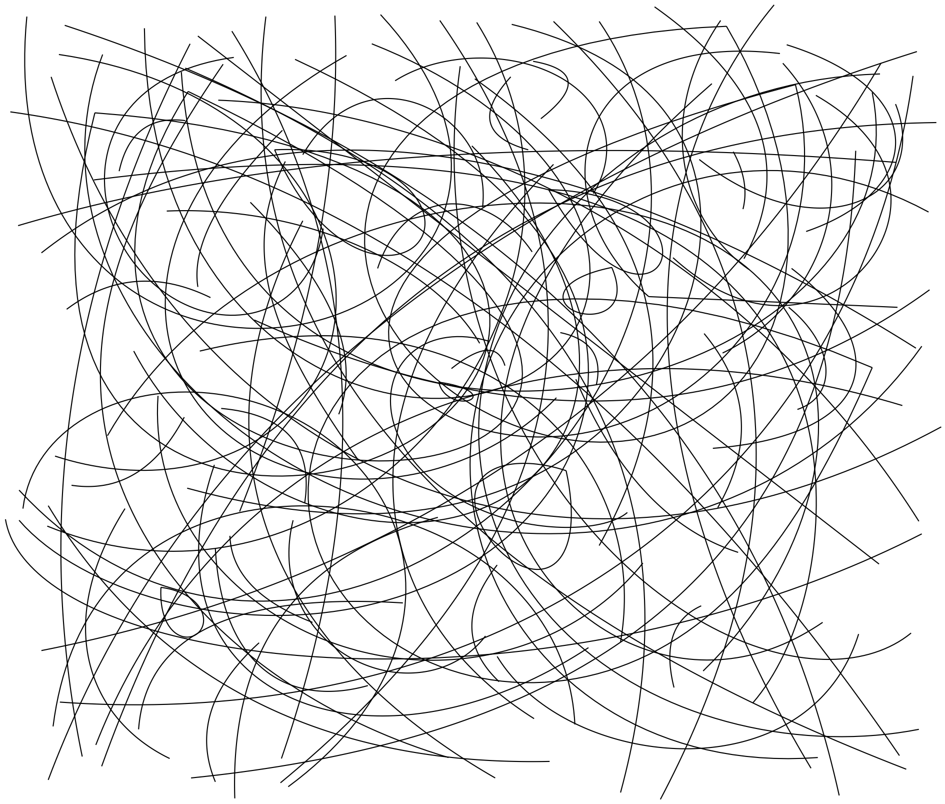


Figure V.23 A *Arrangement de 100 polycourbes à une courbe de Bézier chacune*

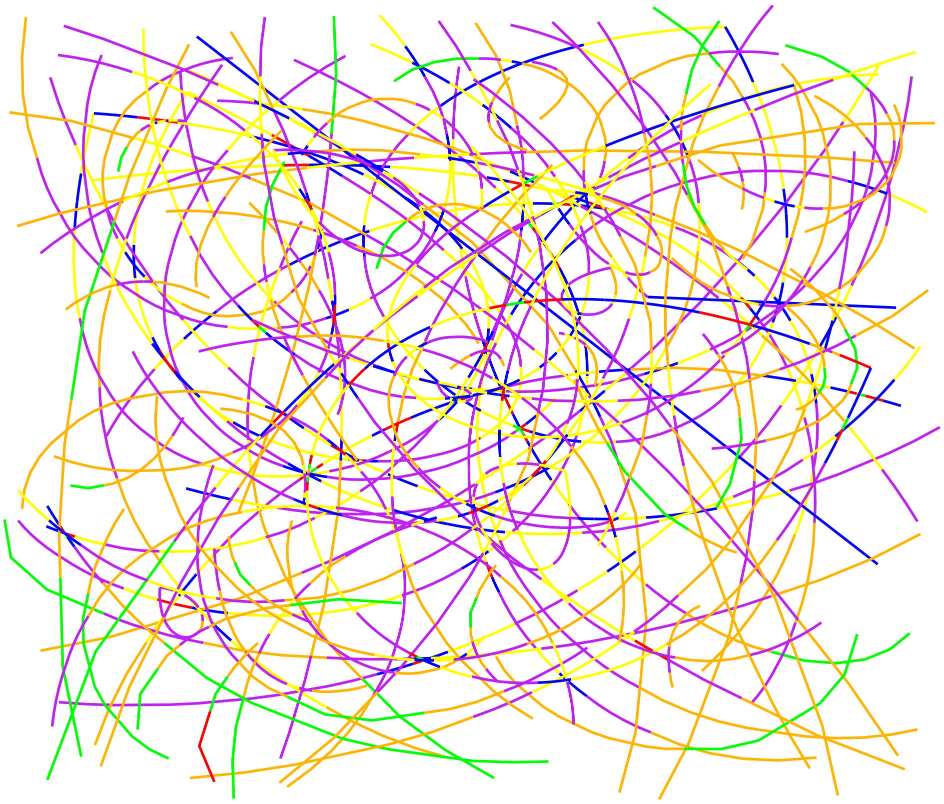


Figure V.24 B *L'arrangement de polygones d contrôle a 7318 sommets*

Figure V.25 C *L'arrangement de polygones support a 1953 sommets*

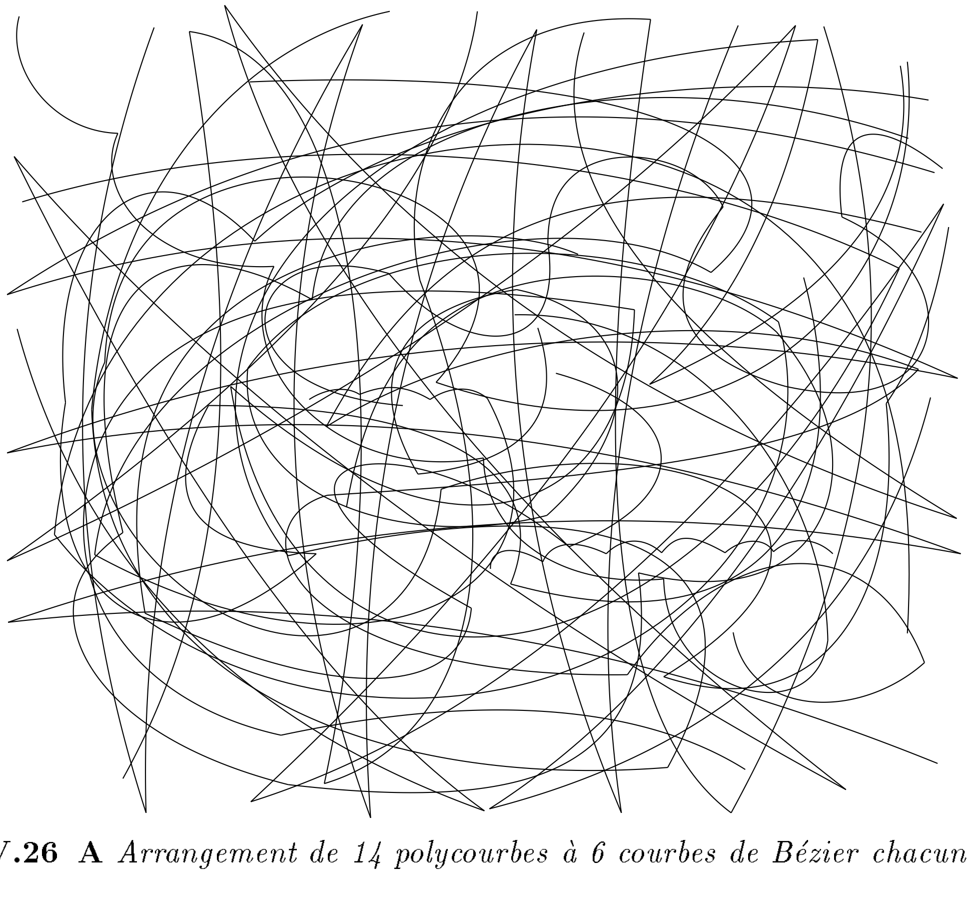


Figure V.26 A *Arrangement de 14 polycourbes à 6 courbes de Bézier chacune*

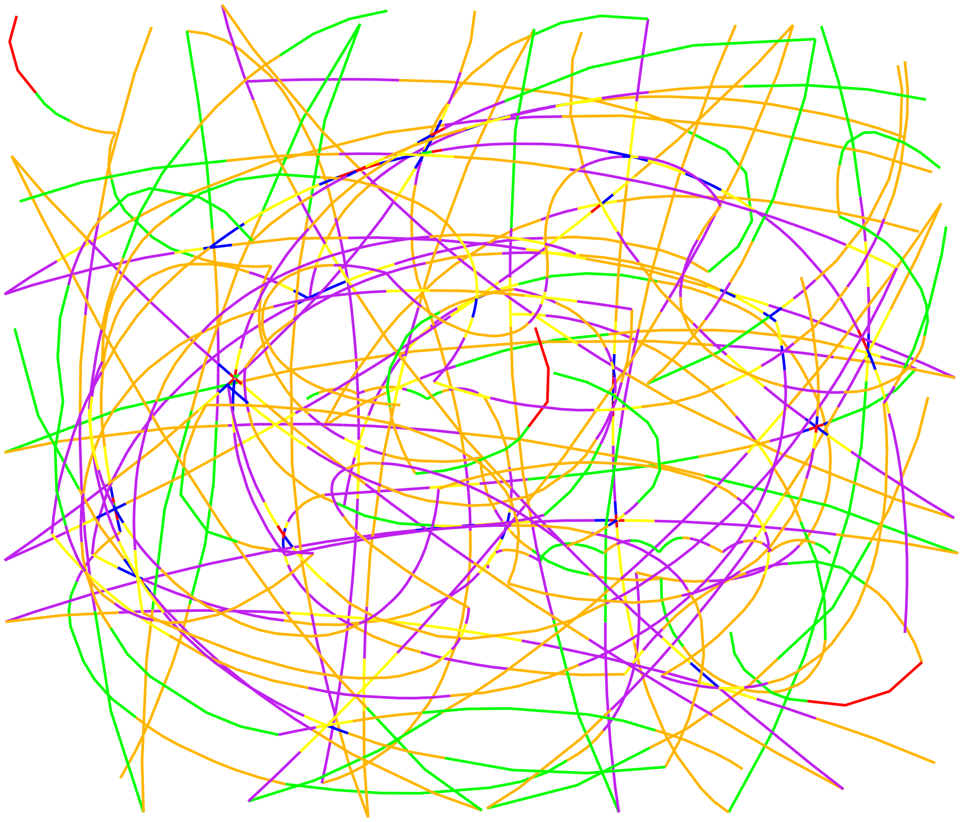


Figure V.27 **B** *L'arrangement de polygones de contrôle a 2370 sommets*

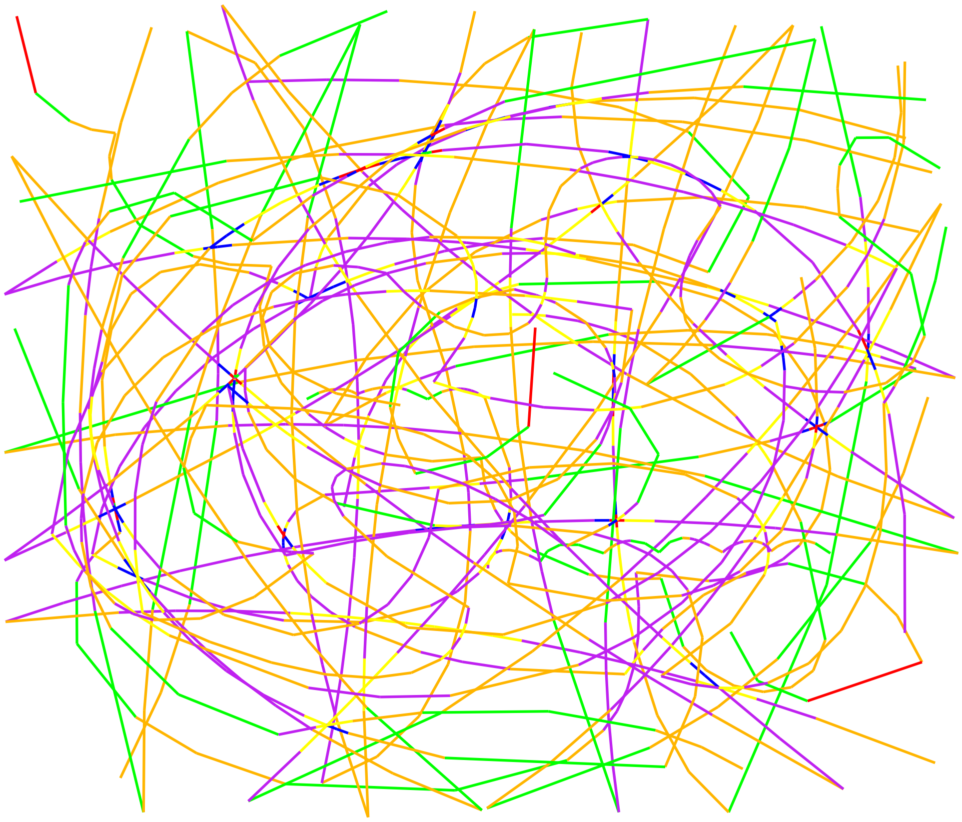


Figure V.28 **C** *L'arrangement de polygones support a 837 sommets*

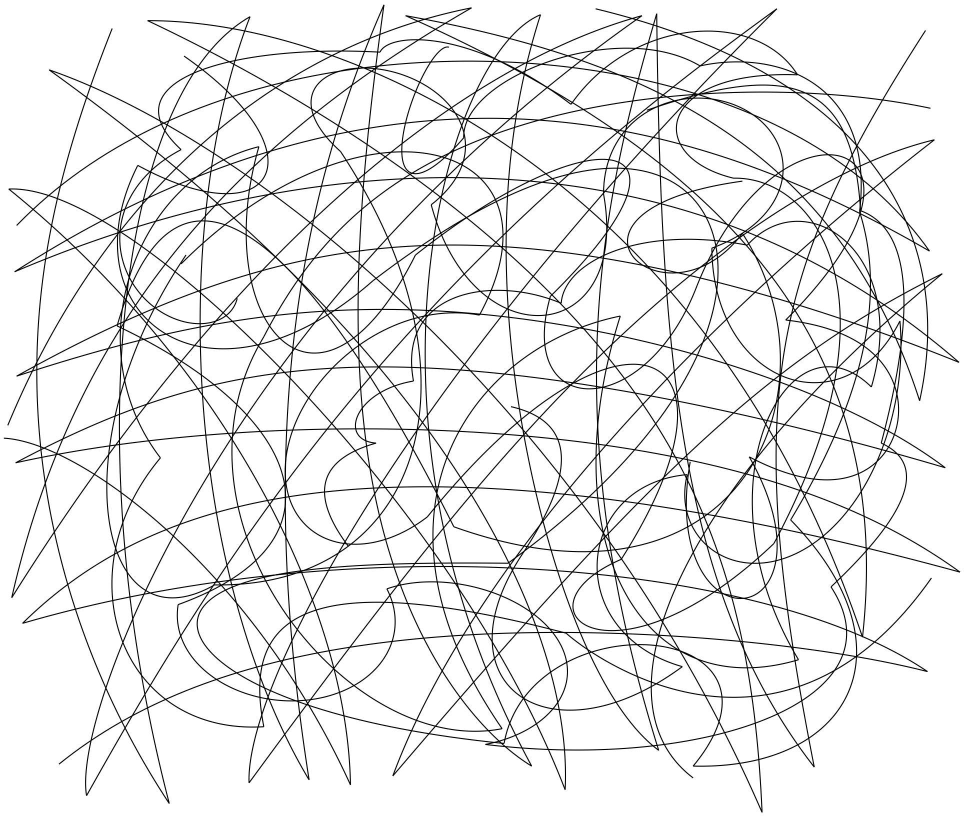


Figure V.29 A *Arrangement de 10 polycourbes à 10 courbes de Bézier chacune*

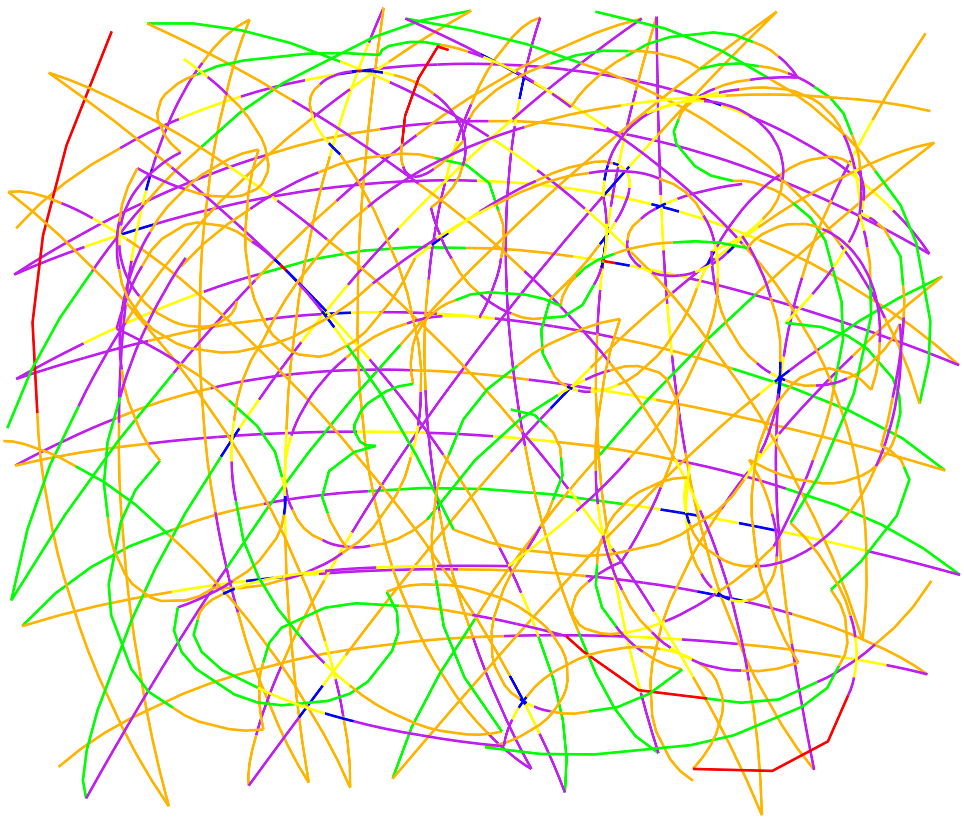


Figure V.30 B *L'arrangement de polygones de contrôle a 2339 sommets*

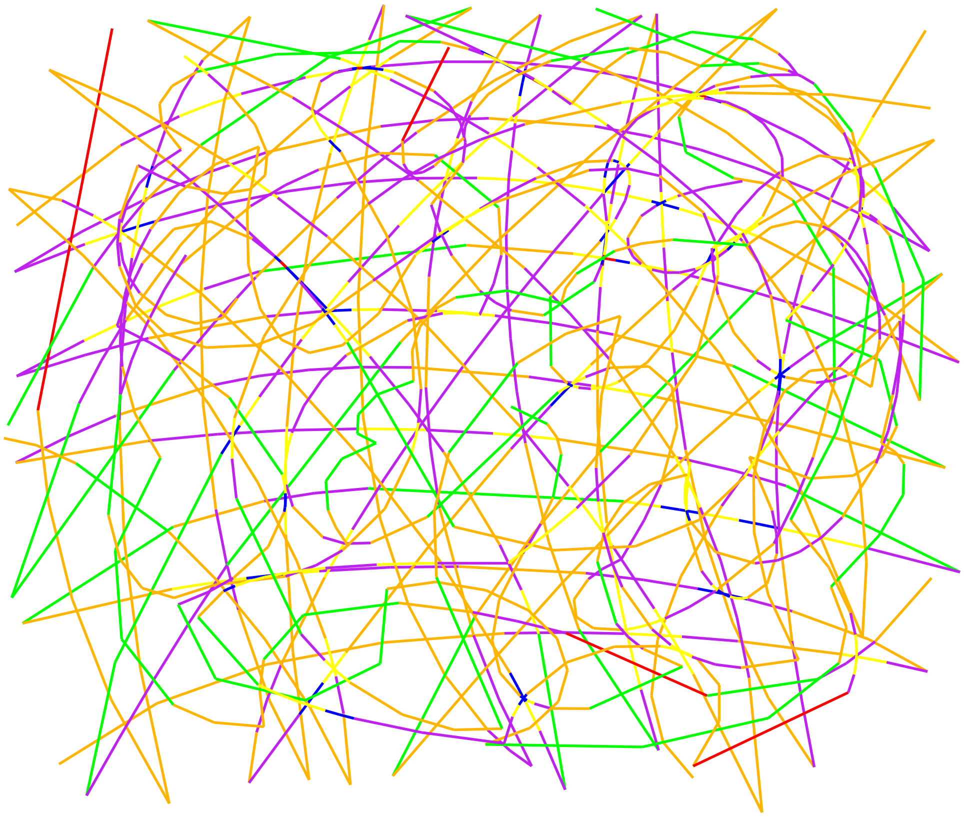


Figure V.31 C *L'arrangenent de polygones support a 716 sommets*

Les trois figures suivantes présentent un arrangement de 27 polycourbes à une courbe de Bézier chacune, sauf une qui en a deux. Les courbes de Bézier ont été introduites de telle manière que le plus d'intersections possible. Ainsi, tous les éléments de l'ensemble $\mathcal{T}$, au nombre de 2935, étaient des triangles simples, mais seulement 23 parmi eux ont demandé des subdivisions pour satisfaire les conditions d'équivalence. Pourtant, ceci a grandement augmenté la taille en courbes de Bézier de l'arrangement : dans sa forme initiale il avait 28 courbes, après le cycle des lignes 12-18 de l'algorithme il avait 142 courbes, pour arriver dans sa forme finale à 280 courbes de Bézier. Le nombre moyen d'itérations nécessaire pour rendre tous les triangles simples équivalents est 1,78, le maximum étant 4.

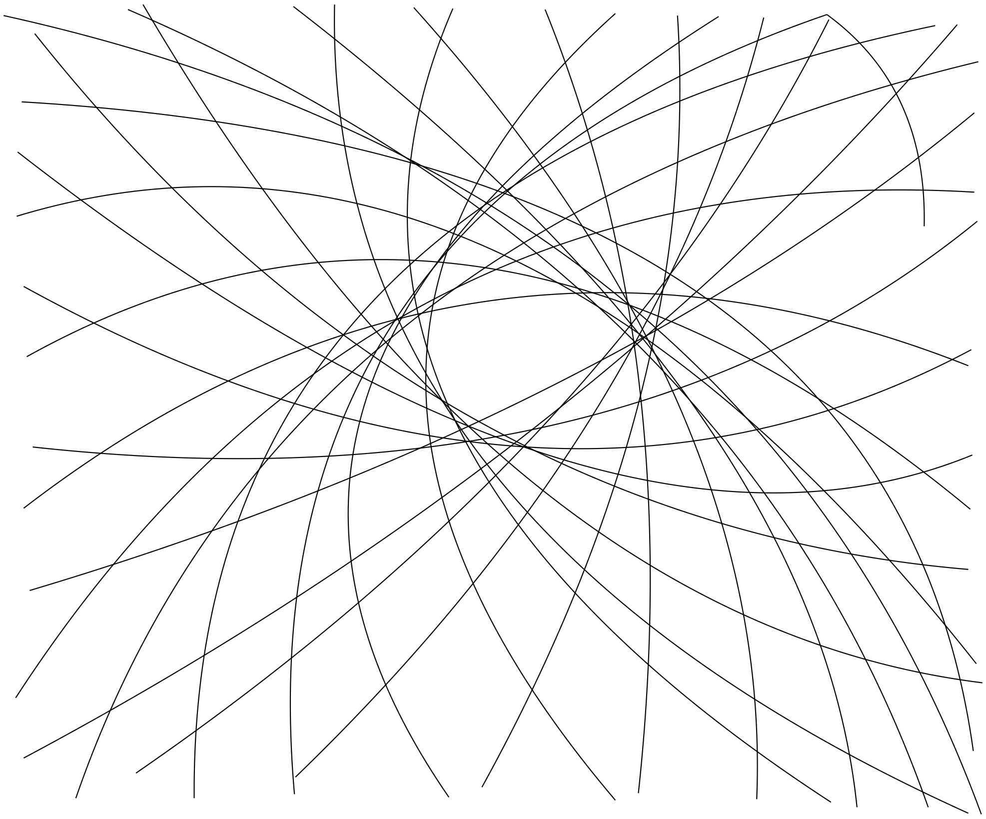


Figure V.32 A

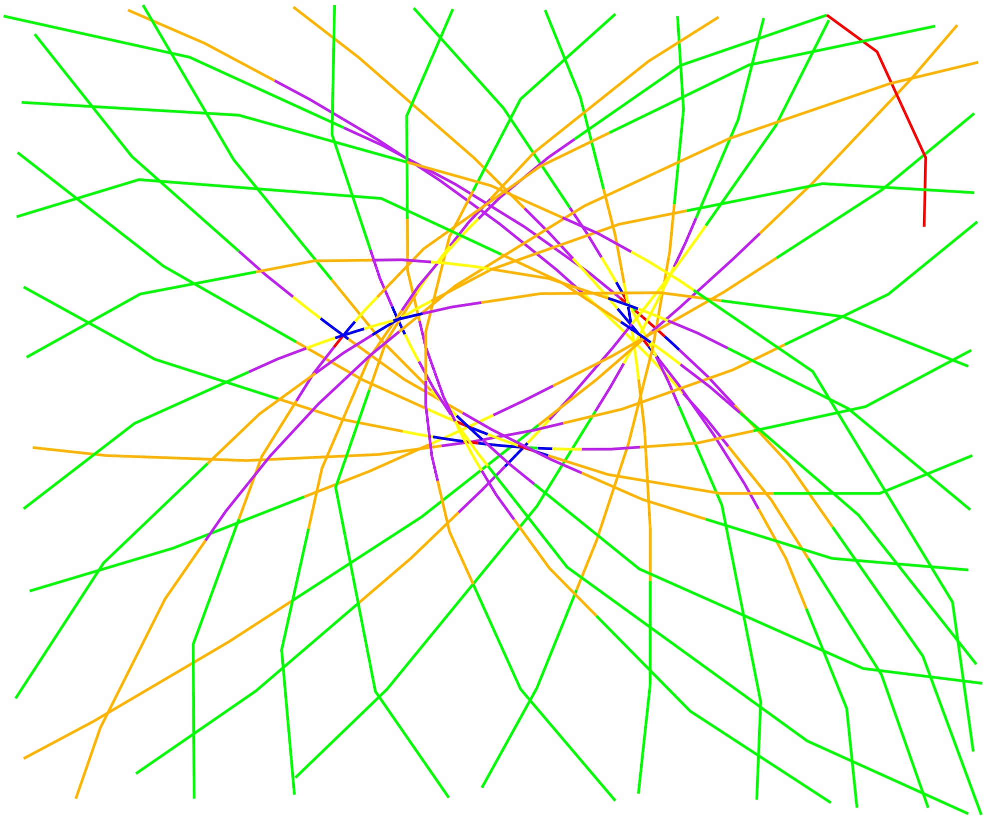


Figure V.33 B *L'arrangement de polygones de contrôle a 587 sommets*

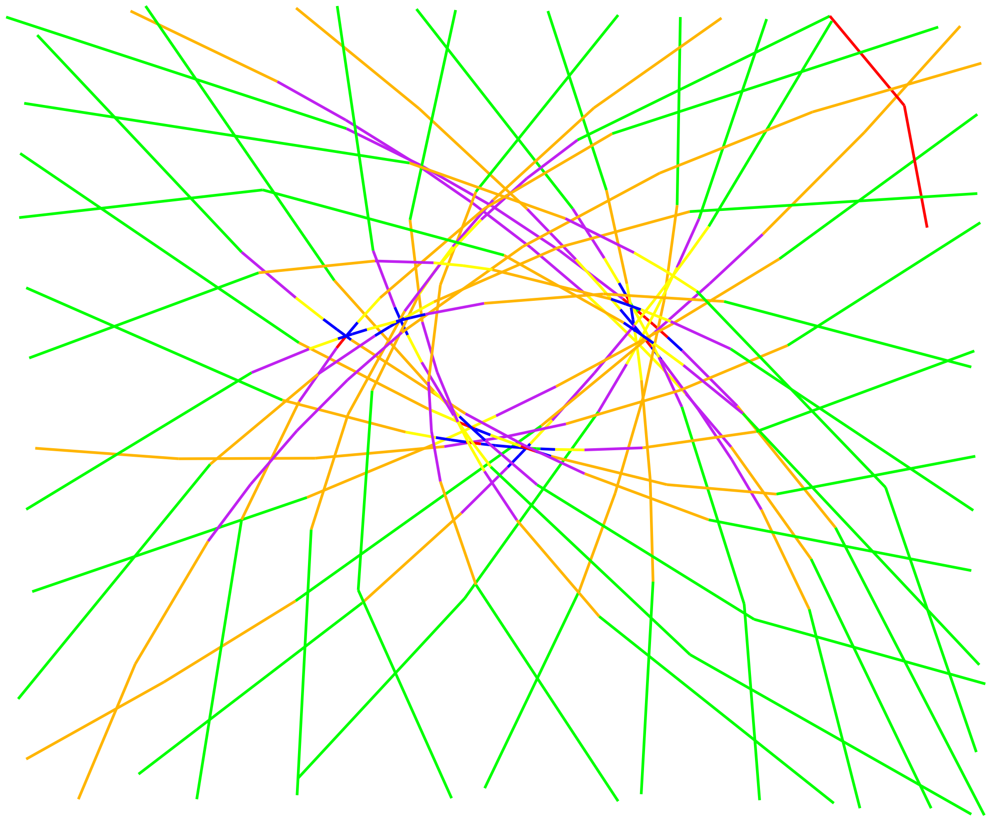


Figure V.34 C *L'arrangement de plygones support a 308 sommets*