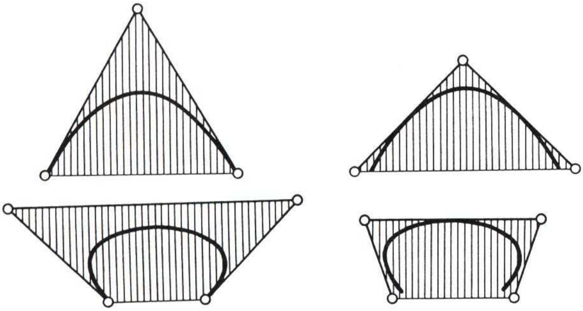

Fig. 12.11. Convex hulls (left) and small convex hulls (right) for quadratic and cubic curves.

It is still faster to construct min-max boxes defined directly from the orner Bézier points. However, to assure that this box entirely encloses the bject, we have to enlarge it appropriately. [Fil 86a] discusses how to find a agnification factor using second derivatives. [Gol 86] gives estimates using e maximal deviation of the curve from the convex hull of its control points.

A rectangular *strip* can be defined as the smallest rectangle containing e convex hull of the Bézier points with one edge parallel to the line $\boldsymbol{b}_0\boldsymbol{b}_n$, ee [Bal 81] and Fig. 12.13. In the quadratic and cubic cases, it is possible to nd still smaller rectangular strips with a small amount of additional effort, ee [Sed 89, 90], [Wan 91a], without losing the property that it is an enclosing eighborhood, see Fig. 12.14.

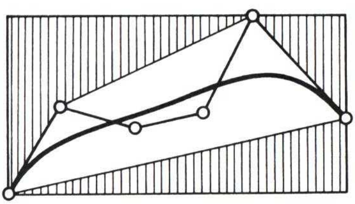

Fig. 12.12. Min-max box for a Bézier curve.

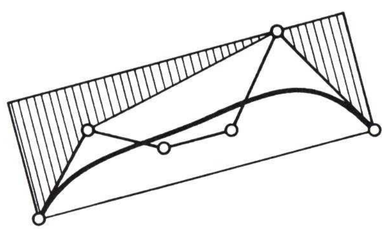

Fig. 12.13. Rectangular strip for a Bézier curve.

Strips of thickness δ (*fat lines*) were investigated for Bézier curves in [Sed 90]. This leads to a very efficient algorithm for finding intersections of curves, where first the Bézier curve $\boldsymbol{X}_1(t)$ is split into pieces using the strip around the second Bézier curve $\boldsymbol{X}_2(\tau)$. Then a new strip is constructed for the middle subsegment of $\boldsymbol{X}_1(t)$, and the clipping process is then applied to the second curve $\boldsymbol{X}_2$, etc. In *Bézier clipping*, subsets of the parameter domain are constructed where it is guaranteed that $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ do not intersect, e.g.,

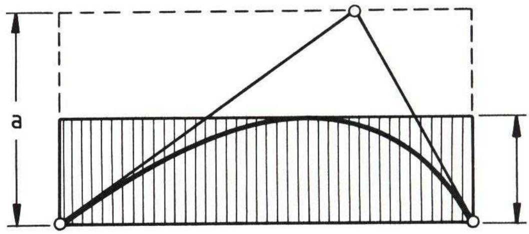

Fig. 12.14a. Reduced rectangular strips for a quadratic Bézier curve.

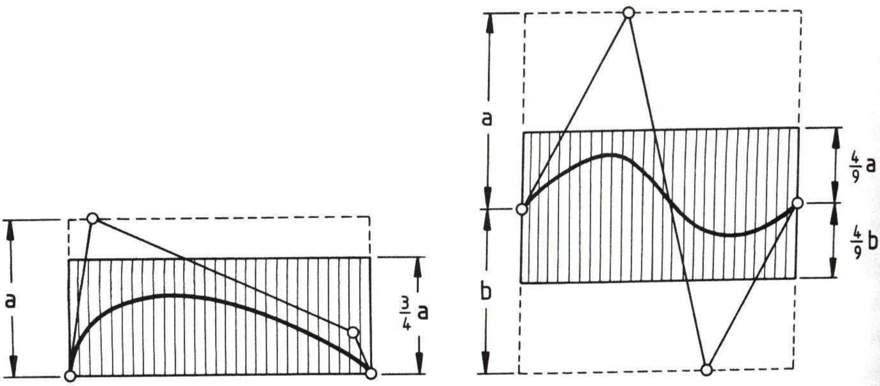

Fig. 12.14b. Reduced rectangular strip for a cubic Bézier curve.

because $\boldsymbol{X}_1$ lies outside of the strip around $\boldsymbol{X}_2$. This can be accomplished as follows: Suppose $\boldsymbol{b}_i = (x_i, y_i)$ denote the Bézier points of $\boldsymbol{X}_1(t)$, and that $\boldsymbol{B}_i = (X_i, Y_i)$ are those of $\boldsymbol{X}_2(\tau)$. Let L be the line joining $\boldsymbol{B}_0\boldsymbol{B}_n$, see Fig. 12.15a:

$$ax + by + c = 0, \qquad \text{with} \qquad a^2 + b^2 = 1.$$

Then the distance $d(t)$ of a point on the curve $\boldsymbol{X}_1(t)$ from L is given by

$$d(t) = \sum_{i=0}^{n} d_i B_i^n(t), \qquad \text{with} \qquad d_i = ax_i + by_i + c.$$

The d_i are the distances of the $\boldsymbol{b}_i$ from L. The function $d(t)$ can be interpreted as a function-valued Bézier curve. To display it graphically, we associate the

Bézier ordinates d_i with the abscissae i/n, see Sect. 4.1 and Fig. 12.15b. The zeros of $d(t) = 0$ correspond to the points where $\boldsymbol{X}_1(t)$ intersects the straight line L. At t's where $d(t) > d_{max}$ or $d(t) < d_{min}$, $\boldsymbol{X}_1(t)$ lies outside of the strip around $\boldsymbol{X}_2(\tau)$, and therefore cannot be a point of intersection with the curve $\boldsymbol{X}_2(\tau)$, see Fig. 12.15a. The points t_{min} and t_{max} where the convex hull of $d(t)$ intersects the strip $d_{min} \leq d \leq d_{max}$ produces a set in the parameter space which is guaranteed not to contain any intersection points of $\boldsymbol{X}_1(t)$ and $\boldsymbol{X}_2(\tau)$. The Bézier clipping process then uses the de Casteljau algorithm to split $\boldsymbol{X}_1(t)$ into three pieces, using $t = t_{min}$ and $t = t_{max}$. Only the middle subsegment of $\boldsymbol{X}_1(t)$ is retained, and its strip is used to split $\boldsymbol{X}_2(\tau)$.

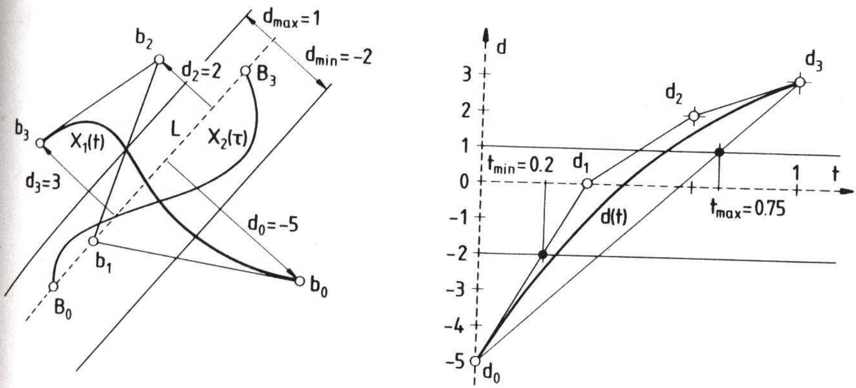

Fig. 12.15a. Bézier curves $\boldsymbol{X}_1, \boldsymbol{X}_2$ and strip for $\boldsymbol{X}_2$.

Fig. 12.15b. Function-valued Bézier curve $d(t)$.

Fig. 12.15. Sederberg-Nishita Bézier clipping algorithm.

When two curves intersect at more than one point, then we must subdivide the curves so that any pair of subsegments has at most one point of intersection. Then we can apply the above algorithm to find each intersection point separately.

A divide-and-conquer algorithm based on *fat planes* of thickness δ was presented in [Car 82] for Bézier surfaces