mance of text searching (and other text extraction operations) is significantly better if the text strings are as long as possible and are shown in natural reading order.

5.3.3 Text Space Details

As stated in Section 5.3.1, "Text-Positioning Operators," text is shown in *text space*, which is defined by the combination of the text matrix, T_m, and the text state parameters T_{fs}, T_h, and T_{rise}. This determines how text coordinates are transformed into user space. Both the glyph's shape and its displacement (horizontal or vertical) are interpreted in text space.

Note: *Glyphs are actually defined in glyph space, whose definition varies according to the font type as discussed in Section 5.1.3, "Glyph Positioning and Metrics." Glyph coordinates are first transformed from glyph space to text space before being subjected to the transformations described below.*

The entire transformation from text space to device space can be represented by a *text rendering matrix*, T_{rm}:

$$T_{rm} = \begin{bmatrix} T_{fs} \times T_h & 0 & 0 \\ 0 & T_{fs} & 0 \\ 0 & T_{rise} & 1 \end{bmatrix} \times T_m \times CTM$$

T_{rm} is a temporary matrix; conceptually, it is recomputed before each glyph is painted during a text-showing operation.

After the glyph is painted, the text matrix is updated according to the glyph displacement and any spacing parameters that apply. First, a combined displacement is computed, denoted by either t_x (in horizontal writing mode) or t_y (in vertical writing mode); the variable corresponding to the other writing mode is set to 0.

$$t_x = \left(\left(w0 - \frac{T_j}{1000}\right) \times T_{fs} + T_c + T_w\right) \times T_h$$

$$t_y = \left(w1 - \frac{T_j}{1000}\right) \times T_{fs} + T_c + T_w$$