Handcraft algorithm

A disjointed algorithm for offsetting cubic Bézier curves

Nicola Fontana

February 25, 2009

This is basically the transcription of the algorithm used by the ADG canvas on its early development phases. This algorithm has been superseded in recent releases by BAIOCA. It is included here for completeness and historical reasons.

Given a cubic Bézier primitive, it must be found the approximated Bézier curve parallel to the original one at a specific distance (the so called "offset curve"). The four points needed to build the new curve must be returned.

To solve the offset problem, a custom algorithm is used. First, the resulting curve MUST have the same slope at the start and end point. These constraints are not sufficient to resolve the system, so I let the curve pass thought a given point (r, known and got from the original curve) at a given time (m, now hardcoded to 0.5).

Firstly, some useful variables are defined:

$$
\begin{aligned}
v_0 &= \text{unitvector}(\text{p}[1] - \text{p}[0]) \times \text{offset} \\
v_3 &= \text{unitvector}(\text{p}[3] - \text{p}[2]) \times \text{offset} \\
p_0 &= \text{p}[0] + \text{normal}(v_0) \\
p_3 &= \text{p}[3] + \text{normal}(v_3)
\end{aligned}
$$

The resulting curve must have the same slopes than the original one at the start and end points. Forcing the same slopes means:

$$p_1 = p_0 + k_0 v_0.$$

where k_0 is an arbitrary factor. Decomposing for x and y:

$$
\begin{cases}
p_{1x} = p_{0x} + k_0 v_{0x} \\
p_{1y} = p_{0y} + k_0 v_{0y}
\end{cases}
$$

and doing the same for the end point:

$$\begin{cases} p_{2x} = p_{3x} + k_3 v_{3x} \\ p_{2y} = p_{3y} + k_3 v_{3y} \end{cases}$$

This does not give a resolvable system though. The curve will be interpolated by forcing its path to pass throught r when *time* is m, where $0 \leq m \leq 1$. Knowing the function of the cubic Bézier:

$$C(t) = (1-t)^3 p_0 + 3t(1-t)^2 p_1 + 3t^2(1-t) p_2 + t^3 p_3.$$

and forcing $t = m$ and $C(t) = r$:

$$\begin{aligned} r &= (1-m)^3 p_0 + 3m(1-m)^2 p_1 + 3m^2(1-m) p_2 + m^3 p_3 \\ (1-m)p_1 + m p_2 &= \frac{r - (1-m)^3 p_0 - m^3 p_3}{3m(1-m)} \end{aligned}$$

brings to the final system:

$$\begin{cases} p_{1x} = p_{0x} + k_0 v_{0x} \\ p_{1y} = p_{0y} + k_0 v_{0y} \\ p_{2x} = p_{3x} + k_3 v_{3x} \\ p_{2y} = p_{3y} + k_3 v_{3y} \\ (1-m)p_{1x} + m p_{2x} = \dfrac{r_x - (1-m)^3 p_{0x} - m^3 p_{3x}}{3m(1-m)} \\ (1-m)p_{1y} + m p_{2y} = \dfrac{r_y - (1-m)^3 p_{0y} - m^3 p_{3y}}{3m(1-m)} \end{cases}$$

Substituting and resolving for k_0 and k_3:

$$\begin{cases} (1-m)k_0 v_{0x} + m k_3 v_{3x} = \dfrac{r_x - (1-m)^3 p_{0x} - m^3 p_{3x}}{3m(1-m)} - (1-m)p_{0x} - m p_{3x} \\ (1-m)k_0 v_{0y} + m k_3 v_{3y} = \dfrac{r_y - (1-m)^3 p_{0y} - m^3 p_{3y}}{3m(1-m)} - (1-m)p_{0y} - m p_{3y} \end{cases}$$

$$\begin{cases} (1-m)k_0 v_{0x} + m k_3 v_{3x} = \dfrac{r_x - (1-m)^2(1+2m) p_{0x} - m^2(3-2m) p_{3x}}{3m(1-m)} \\ (1-m)k_0 v_{0y} + m k_3 v_{3y} = \dfrac{r_y - (1-m)^2(1+2m) p_{0y} - m^2(3-2m) p_{3y}}{3m(1-m)} \end{cases}$$

Letting:

$$s = \frac{r - (1-m)^2(1+2m)p_0 - m^2(3-2m)p_3}{3m(1-m)}$$

reduces the above to this final equations:

$$\begin{cases} s_x = (1-m)k_0 v_{0x} + mk_3 v_{3x} \\ s_y = (1-m)k_0 v_{0y} + mk_3 v_{3y} \end{cases}$$

If $v_{0x} \neq; 0$, the system can be resolved for k_0 and k_3 calculated accordingly:

$$\begin{cases} k_0 = \dfrac{s_x - mk_3 v_{3x}}{(1-m)v_{0x}} \\ s_y = \dfrac{(s_x - mk_3 v_{3x})v_{0y}}{v_{0x}} + mk_3 v_{3y} \end{cases}$$

$$\begin{cases} k_0 = \dfrac{s_x - mk_3 v_{3x}}{(1-m)v_{0x}} \\ s_y - \dfrac{s_x v_{0y}}{v_{0x}} = k_3 m(v_{3y} - \dfrac{v_{3x} v_{0y}}{v_{0x}}) \end{cases}$$

$$\begin{cases} k_0 = \dfrac{s_x - mk_3 v_{3x}}{(1-m)v_{0x}} \\ k_3 = \dfrac{s_y - s_x \frac{v_{0y}}{v_{0x}}}{m(v_{3y} - v_{3x}\frac{v_{0y}}{v_{0x}})} \end{cases}$$

Otherwise, if $v_{3x} \neq 0$, the system can be solved for k_3 and k_0 calculated accordingly:

$$\begin{cases} k_3 = \dfrac{s_x - (1-m)k_0 v_{0x}}{mv_{3x}} \\ s_y = (1-m)k_0 v_{0y} + \dfrac{[s_x - (1-m)k_0 v_{0x}]v_{3y}}{v_{3x}} \end{cases}$$

$$\begin{cases} k_3 = \dfrac{s_x - (1-m)k_0 v_{0x}}{mv_{3x}} \\ k_0(1-m)(v_{0y} - k_0 v_{0x}\dfrac{v_{3y}}{v_{3x}}) = s_y - s_x\dfrac{v_{3y}}{v_{3x}} \end{cases}$$

$$\begin{cases} k_3 = \dfrac{s_x - (1-m)k_0 v_{0x}}{mv_{3x}} \\ k_0 = \dfrac{s_y - s_x \frac{v_{3y}}{v_{3x}}}{(1-m)(v_{0y} - v_{0x}\frac{v_{3y}}{v_{3x}})} \end{cases}$$

The whole process must be guarded against division by 0 exceptions. If either v_{0x} and v_{3x} are 0, the first equation will be inconsistent. More in general, the $v_{0x} \times v_{3y} = v_{3x} \times v_{3y}$ condition must be avoided. This is the first situation to avoid, in which case an alternative approach should be used.