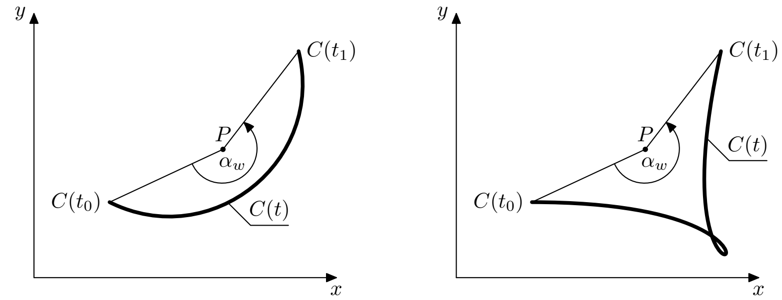
ON COMPUTING A WINDING NUMBER FOR BÉZIER SPLINES

Bogusław Jackowski

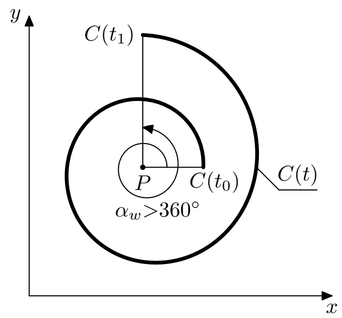
Assume that we have given a point P in the plane and the planar curve $C(t)$ defined for $t_0 \le t \le t_1$. The total angle encircled by the radius $PC(t)$ as t runs from t_0 to t_1 we will call the *winding angle* and denote by α_w:



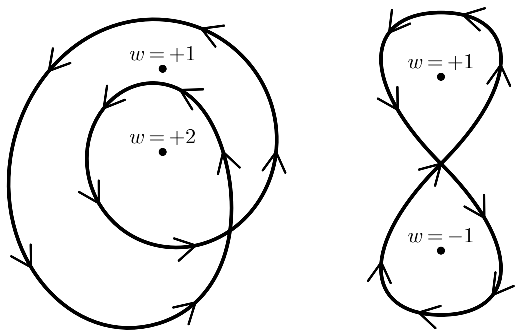
Note that the winding angle is insensitive to certain local properties of the curve $C(t)$ (e.g., local loops): in the figures above, the *winding* angle is the same in both cases (it is assumed that points P, $C(t_0)$ and $C(t_1)$ coincide).

The winding angle is positive if the the point P lies to the right with respect to the point traversing the curve, and negative otherwise.

Of course, the absolute value of a winding angle can be larger than 360°:



For cyclic curves, the *winding angle* is always a multiple of 360°, i.e., $\alpha_w = 360°w$, where w is an integer. The entity w is called the *winding number* (for a given point and curve).



In the sequel, we will focus our attention on cylic Bézier splines.

The idea of the algorithm computing the *winding number* for Bézier splines is due to Laurent C. Siebenmann (metafont@ens.fr, 2000; now the MetaPost Discussion List is hosted by TUG—metapost@tug.org). Siebenmann's solution, however, was MetaPost-oriented—it exploited heavily the operation *arctime*, available in MetaPost but unavailable, e.g., in MetaFont. Below, I'll present an algorithm basing on the same idea but referring to more elementary properties of a Bézier segment.

For a given point P and a Bézier spline C, we will try to find the *winding angle* by measuring the *winding angles* for a discrete series of time points. First, we will try to measure angles between nodes 0, 1, 2, …, n of the spline C. If the relevant Bézier segments are appropriately short, the sum of the angles yields the total *winding angle*. The problem arises, when the Bézier arc is too long—see, e.g., the leftmost panel of the first figure (the angle $C(t_0)PC(t_1)$ equals $360° - \alpha_w$).

The main observation of Siebenmann is as follows: if the length of the subarc $C(t)$ for $t_0 \le t \le t_1$ is shorter than the length of the longer of the radii $PC(t_0)$ and $PC(t_1)$, than we can safely assume that the (acute) angle between $PC(t_0)$ and $PC(t_1)$ is the *winding angle*. Actually, we do not need to know the exact length of the arc—an approximation suffices. If B_a, B_b, B_c, and B_d are points defining a Bézier arc B (i.e., B_a and B_d are its nodes, B_b and B_c are its control points), then

$$|B_aB_b| + |B_bB_c| + |B_cB_d| \ge |B|$$

($|\ldots|$ denotes the length of an interval and the length of a Bézier arc). In other words, we can safely use the left-hand side of the above inequality instead of the true value of the arc length in the computation of the *winding angle/number*.

The algorithm can be expressed in a "pseudocode" as follows:

input: a point P and a Bézier spline B, consisting of segments $B_1, B_2, \ldots, B_n$
output: α_w – the winding angle for P and B
procedure *windingangle*(P, B)
  if B is a single segment
    let B_a, B_b, B_c, B_d be the consecutive control nodes of the segment B
    if $\min(|PB_a|, |PB_d|) <$ *assumed minimal distance*
      exit (P almost coincides with B, winding angle incalculable)
    fi
    if $|B_aB_b| + |B_bB_c| + |B_cB_d| > \max(|PB_a|, |PB_d|)$
      return *windingangle*$(P, B(0, 1/2))$ + *windingangle*$(B(1/2, 1))$
    else
      return angle α between the radii PB_a and PB_d $(-90^\circ < \alpha < 90^\circ)$
    fi
  else
    return *windingangle*$(P, B_1) + \ldots +$ *windingangle*(P, B_n)
  fi
end

An example of MetaPost/MetaFont implementation is given below:

```
vardef mock_arclength(expr B) = % |B| -- B\'ezier segment
 % |mock_arclength(B)>=arclength(B)|
 length((postcontrol 0 of B)-(point 0 of B)) +
 length((precontrol 1 of B)-(postcontrol 0 of B)) +
 length((point 1 of B)-(precontrol 1 of B))
enddef;

vardef windingangle(expr P,B) = % |P| -- point, |B| -- B\'ezier spline
 if length(B)=1: % single segment
  save r,v;
  r0=length(P-point 0 of B); r1=length(P-point 1 of B);
  if (r0<2eps) and (r1<2eps): % MF and MP are rather inaccurate, we'd better stop now
   errhelp "It is rather not advisable to continue. Will return 0.";
   errmessage "windingangle: point almost coincides with B\'ezier segment (dist="
      & decimal(min(r0,r1)) & ")";
   0
  else:
   v:=mock_arclength(B); % |v| denotes both length and angle
   if (v>r0) and (v>r1): % possibly too long B\'ezier arc
    windingangle(P, subpath (0, 1/2) of B) + windingangle(P, subpath (1/2, 1) of B)
   else:
    v:=angle((point 1 of B)-P)-angle((point 0 of B)-P);
    if v>=180: v:=v-360; fi  if v<-180: v:=v+360; fi
    v
   fi
  fi
 else: % multisegment spline
  windingangle(P,subpath (0,1) of B)
   for i:=1 upto length(B)-1: + windingangle(P,subpath (i,i+1) of B) endfor
 fi
enddef;
```

Note that although the returned angle (line 23 in the MF/MP code above) is acute, the difference of the component angles (line 21) can be ouside the interval $\langle -180°, 180° \rangle$; hence the normalization (line 22).

If the operation *windingnumber* is needed for some reasons, it can be implemented trivally:

```
vardef windingnumber (expr P,B) = % |P| -- point, |B| -- B\'ezier spline
 windingangle(P,B)/360
enddef;
```

The operations *windingangle* or, equivalently, *windingnumber* can be used, e.g., for determining the mutual position of two nonintersecting cyclic curves (whether one is embeded inside the other or not):

```
tertiarydef a inside b =
 if path a: % |and path b|; |a| and |b| must be cyclic and must not touch each other
  begingroup
   save a_,b_; (a_,b_)=(windingnumber(point 0 of a,b), windingnumber(point 0 of b,a));
   (abs(a_-1)<eps) and (abs(b_)<eps)
  endgroup
 else: % |numeric a and pair b|
  begingroup
   (a>=xpart b) and (a<=ypart b)
  endgroup
 fi
enddef;
```

Gdańsk, April–May, 2011

Postscriptum. In some cases, another definition, equivalent to the one formulated above may be useful (the formulation, given below without a proof of equivalence, is a slightly edited excerpt from the Laurent C. Siebenmann's email):

Assume that there are given curve C and point P. Choose at random a line segment emanating from the point P to the point W, with W outside the bounding box of C and P. Inductively examine the intersection points Q of PQ with C. Supposing these points Q are all "nondegenerate" intersections, they are also finite in number, and a sign $+1$ or -1 is associated to each. Nondegenerate means that Q is a smooth point of c and the tangent vector T to C at Q is not parallel to PQ, and that Q is not a point where C crosses itself. The sign to use is the sign of the wedge product '$(Q - P)$ wedge T', i.e.,

$$(Q - P) \cdot (T \text{ rotated } -90)$$

The sum of the signs is the *winding number*.

It is a probabilistic theorem that degenerate intersections will rarely be met.