LOW LEVEL DIRECT INTERPOLATION FOR PARAMETRIC CURVES

Ruiz OSCAR (Professor)[♦]
Martinez ADRIANA (Research Assistant)
Rendon ELIZABETH (Student Assistant)
oruiz@sigma.eafit.edu.co
Center for Interdisciplinary Research (CII) in CAD / CAM / CG
EAFIT University
Medellin, COLOMBIA

Abstract

We present an algorithm for the direct interpolation of parametric curves with a CNC machine. The algorithm expresses parametric planar curves as sequences of discrete axes movements of BLU size of the machining tool. Therefore, the curve C(u) is directly approximated by the pulse trains, hence eliminating one source of the machining errors.

Keywords: Direct Numerical Control, Parametric curve interpolation.

1. INTRODUCTION

In CNC the parametric curve or surface actually machined differs from the ideal mathematical entity. Generation of tool locus for machining of parametric planar curves usually presents two levels of approximation. The first level is implied by the aproximation of the parametric curve into a sequence of lower order curves (lines, circular arcs, parabolic segments), which are the usual primitives included in the dictionary of a CNC machine. The second level of approximation lies on the fact that these primitives are actually mathematical abstractions which are expressed as a sequence of elementary, discrete movements of the machine tool axes. These two aproximations eventually add up, increasing the machining errors. According to these facts, this investigation presents an algorithm that directly expresses parametric planar curves as sequences of axis movements. This problem can be abstracted as one of approaching a continuous curve in a discrete space. By avoiding intermediate steps, the errors inherent to both approximations are reduced, therefore producing the best possible curve given a CNC precision range (or BLU). The objective of this investigation is to explore the expression of parametric planar curves directly into discrete movements of BLU size of the machining tool. The net final goal is to introduce a new G primitive, namely parametric 2D curve, to enrich the vocabulary of the G code in a CNC machine tool.

[♦] Published in REVISTA UNIVERSIDAD EAFIT, No. 110 April-June, 1998

The example presented interpolates bezier curves, although the algorithm is designed to work with any type of parametric curve. Quantification of the merits of the algorithm is attempted, by comparing possible errors incurred by the traditional algorithms and the one proposed here. Section 2 of the article surveys the existing literature on this topic. Section 3 presents the algorithm and its different characteristics. Section 4 discusses the results obtained in the examples. Section 5 draws the general conclusions of the article, and states possible advances in the topic presented.

2. LITERATURE SURVEY

The general problem of approaching a continuous planar curve by a series of discrete positions varies according to whether the curve is expressed in (i) implicit or (ii) parametric form. (i) For implicit curves there is an immediate antecedent in the algorithms used to display geometric primitives on discrete spaces. This is the case of Bresenham´s and Midpoint algorithms to draw straight lines and circles on raster graphics devices as sequences of screen pixels (Joy). These algorithms make use of symmetries of the primitive to reduce calculations and accelerate the determination of the pixels to highlight. For example, in a circle, the complete set of (x,y) pixel positions forming the perifery can be inferred from the pixels of one octant. Another optimization of the algorithms regards the statement of formulae that only make use of integer arithmetic. Clearly, for general planar curves, the special conditions that favor Bresenham´s and Midpoint algorithms are not present: in general, there exists neither symmetry nor formulae manipulation for integer arithmetic. In addition to software based procedures (Bresenham´s and Midpoint), hardware ones are available which perform the interpolation of an implicit planar funcion by using arrays of DDAs (YOR.76). In summary this scheme obtains the goal function by integrating its derivatives by hardware. A similar software approach, called pattern recognition tracks a continuous planar implicit curve by staircase displacementes on a discrete grid [Kappor notes].
(ii) In processing line drawings (image consisting of line and /or curve segments which need no to be connected) , (TOU, FRE.74) one approaches the given mathematical curve using the sequence of the grid discrete points. The curve points on the grid are chosen on the basis off how the line drawing intersect the grid line between two adjacent mesh nodes . Of the two mesh nodes , the one closer to the intersection point is chosen and used for the low level direct interpolation.

A large obstacle to apply the schemes mentioned above is the fact that they address implicit planar curves. Parametric planar curves are not always convertible to implicit form, and therefore the techniques are not easily applicable. This investigation proposes a method to approach a parametric curve by choosing a sequence of vertex on a grid in such a way that the error of approximation is minimized among them. The algorithm correctly approaches bezier as well as spline curves, and its strategy is indifferent to the type of curve used.

3. METHODOLOGY

Given a planar parametric curve C(u), and a discretization of the 2D space (a grid of size BLU[1]), the goal is to produce the sequence of grid intersection points that approaches C(u). The proposed algorithm obtains the best approach for C(u) given a grid G. This is so, because the algorithm choses in each step the grid intersection that is closest to the curve C(u). Therefore, the error of approximation to the curve is only the inherent to the finite resolution of the machine tool. This is in contrast with other approaches which add the error inherent to the approximation of the curve C(u) by other set of primitives (lines, circles and paraboles).

The algorithm used for the SPLINE interpolation basically includes the following functions:

```
function interp_spline(     Curve C, Grid G,
                            List X, List Y,
                            List Pulses_X, List Pulses_Y)

        initialize(C,G,du);
        X=[]
        Y=[]
        Pulses_X=[]
        Pulses_Y=[]
        (xi, yi) = attract(C(0),G)
        u=0
        while (u  <= 1.0 ) do
        {Inv: (xi,yi) = last grid intersection found}
                u = u + du
                (xt, yt)=C(u)
                if too_large_jump((xi,yi),G,(xt,yt))
                        u=u-du
                        u=du/2
                elseif hits_grid((xi,yi),G,(xt,yt))
                        (xt,yt) = round((xt,yt),G)
                        Pulses_X = [Pulses_X, sign_with_error(xt-xi,ERROR_DIST)]
                        Pulses_Y = [Pulses_Y, sign_with_error(yt-yi,ERROR_DIST)]
                        xi = xt
                        yi = yt
                        X=[X , xi]
                        Y=[Y , yi]
```

[1] BLU: Basic Lenght Unit, is the axis resolution in a CNC machine tool.

```
23              fi
24          end{while}
end{function}

function boolean too_large_jump((xi,yi),G,(xt,yt))
1       if ((|xt-xi| > G) or (|yt-yi| > G) and
2           ( not equal(xt,xi,ERROR_DIST)) and
3           ( not equal(yt,yi,ERROR_DIST))
4          )
5               return( TRUE );
6       else
7               return( FALSE );
8       fi
end{function}

function hits_grid((xi,yi),G,(xt,yt))
1       if (    (equal(|xt-xi|,G,ERROR_DIST)) or
2               (equal(|yt-yi|,G,ERROR_DIST))
3          )
4               return( TRUE );
5       else
6               return( FALSE );
7       fi
end{function}
```

Observations:

1. The function sign_with_error(v, ERROR_DIST) returns

   0, if equal(v,0,ERROR_DIST)
   1, if $v > 0$
   -1, if $v < 0$

   The sequences so build, Pulses_X and Pulses_Y, are the inputs for the step motors driving the axes X and Y of the CNC machine tool.

2. The function initialize(C,G,du) assigns a starting value to du, based on the length of the curve C and the grid size G. The starting value clearly depends on the control points of the curve G, and is defined to permit several iterations on the parameter u within each grid interval.

3. The function too_large_jump((xi,yi),G,(xt,yt)) examines if xt (or yt) differs from xi (or yi) by a distance strictly larger than G () (line 12). In that case the parameter du is decreased,

given the fact that it represents

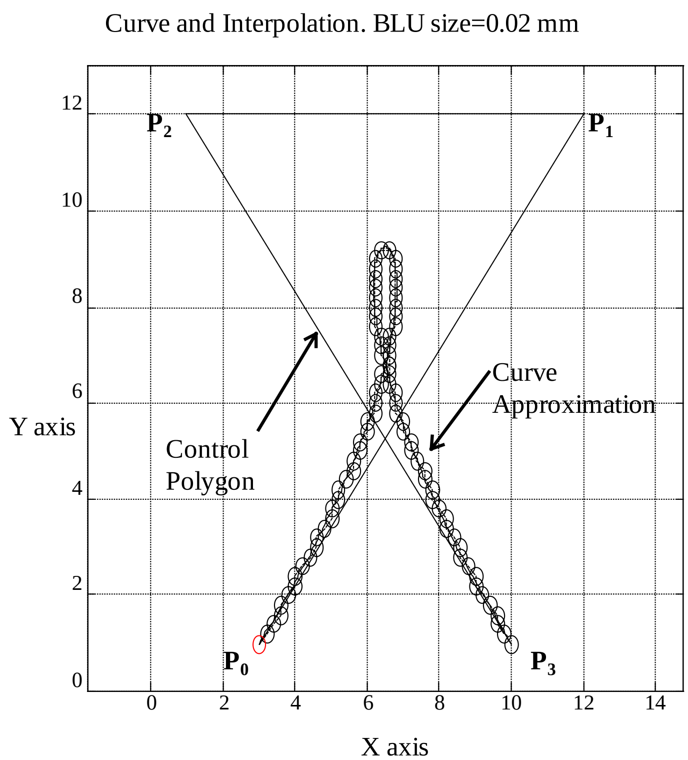

Fig.1

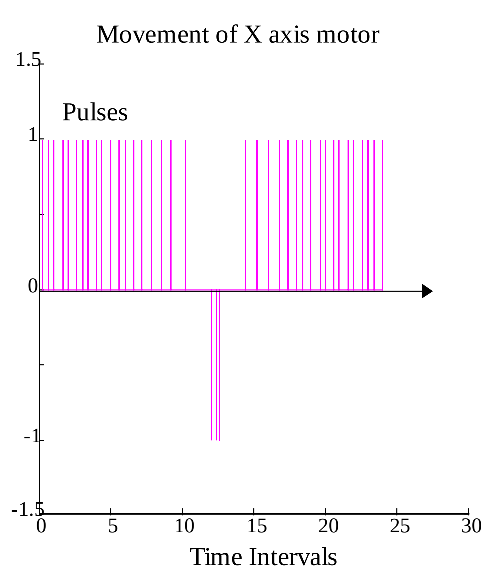

Fig 2. Secuence of pulses X axis motor for curve in Fig. 1

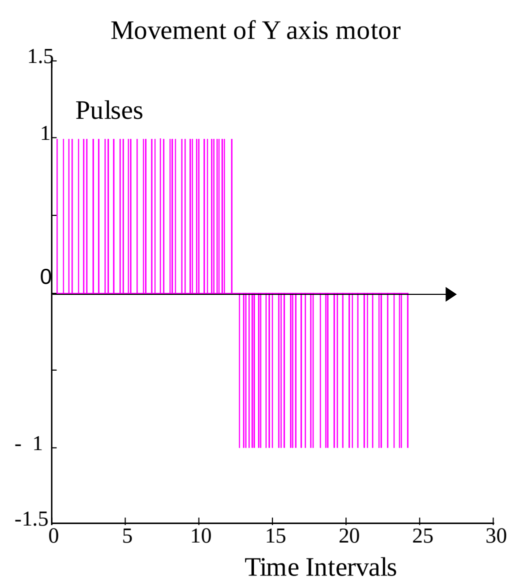

Fig 3. Secuence of pulses Y axis motor for curve in Fig. 1

4. The function hits_grid((xi,yi),G,(xt,yt)) determines whether xt (or yt) lies away from xi (or yi) approximately by a distance G.

5. The line (xt,yt) = round((xt,yt),G) choses the grid intersection closest to the point C(u). This function is called when C(u) lies on a horizontal grid line (y=yt) or a vertical one (x=xt).

The main strategy of the algorithm is to detect the places in which the curve C(u) crosses the vertical (x=k*g) or horizontal (y=k*g) grid lines (see line 15). If it is detected that the curve crosses say a horizontal grid line (yt=k*g, for some integer k), the other component (xt) of the point is attracted to the closest vertical grid line. A converse situation is produced if the line crosses a horizontal grid line.

Once the grid intersection is recorded, the pulses required to drive the machine from the previous intersection to the current one are determined (lines 17,18). Each entry of the pulse train may take one of tree values: {no pulse, pulse forward, pulse backward} (0,1,-1). With this convention, the curve C(u) is represented by two sequences:

for X axis: 1,1,0,0,0,0,-1,-1,-1....
for Y axis: 0,0,-1,-1,-1,0,-1,0,0,0,....

4. RESULTS

The algorithm presented above was tested with three Bezier curves. Figures 1, 4 and 7 present the mathematic versions, as well as the approximations resulting of the algorithm explained. Figures
2, 3, 5, 6, 8, and 9 present the corresponding pulse trains to machine the curves with a CNC machine of 0.02 mm BLU.

5. CONCLUSIONS

The present article has presented an algorithm for the direct interpolation of parametric curves with
a CNC machine.The exact nature of the curves is not relevant to the algorithm given the fact that its mathematical form does not explicitely appear in it. The algorithm presented is the best approximation that a parametric curve may have with a grid of given BLU value. This is so because
it eliminates the approximation of the C(u) curve with given primitives (lines, circles, etc) which have to be, in turn, approached by the pulse trains driving the axes of the CNC machine. In this algorithm the curve C(u) is directly approximated by the pulse trains, therefore eliminating one source of the machining errors.

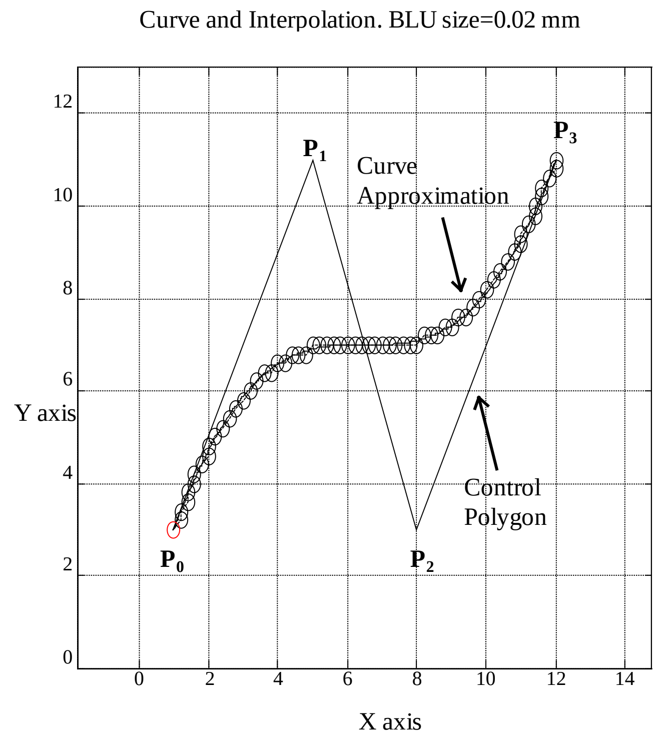

Fig .4

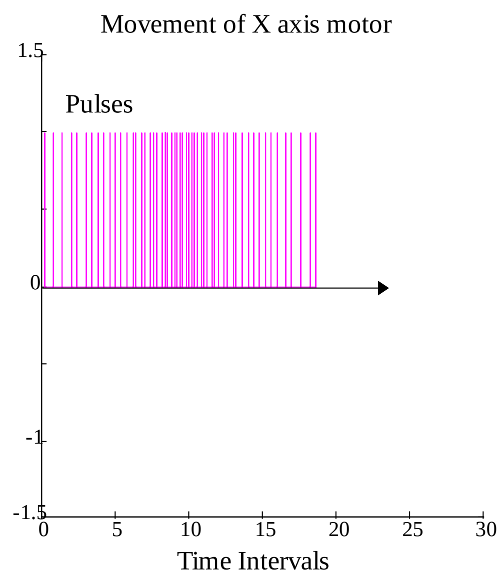

Fig 5. Secuence of pulses X asis motor for curve in Fig. 4.

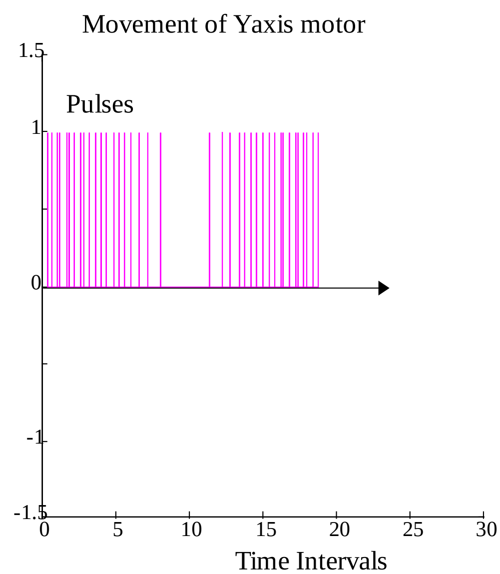

Fig 6. Secuence of pulses Y asis motor for curve in Fig. 4

REFERENCES

[WON.95] WONG Y.S., H.T LOH , C.S. TENG. "*An optimal PieceWise Curve-Fitting Approach for CNC Machining* " ,Transactions of NAMRI/SME, Vol 23, 1995, pp. 157-162.

[FRE.74] H. FREEMAN. "*Computer processing of line-drawing images*", Computing Surveys, Vol. 6, No. 1,March 1974, pp. 58-97.

[FRE.69] H. FREEMAN , J.M. GLASS, "*On the quantization of line-drawing data*", IEEE Transactions on System Science and Cybernetics, Vol. SSC-5, January 1969, pp. 70-79.

[TOU] G.T. TOUSSAINT. "*Processing line drawings*", pp. 48-62.

[KAP] KAPPOR. Internal notes.Univesity of Illinois, Department of Mechanical and Industrial Engineering.

[YOR.76] KOREN YORAM. "*Computer Control of Manufacturing Systems*", Mc Graw-Hill,1976, pp.1-25.

[JOY] KENNETH I. JOY ."*Bresenham´s Algorithm* ", On-line Computer Graphics Notes, Department of Computer Science. University of California, Davis. pp.1-15

[FOL.91] FOLEY, VAN DAM, FEINER ,HUGHES. "*Computer Graphics: Principles and Practice*", Addison Wesley, 1991,pp. 73-78.

[HAN.96] WENZ HANS J. "*Interpolation of Curve Data by Blended Generalized Circles*", Computer Aided Geometric Design, Vol. 13, Issue 8, 1996, pp. 673-680.

[FAR.95] G. FARIN , D. JUNG . "*Linear Precision of Rational Bezier Curves*", Computer Aided Geometric Design, Vol. 12, Issue 4, 1995, pp. 431-433.

[MOR.85] MORTENSON, M. "*Geometric Modeling*". John Wiley and Sons, New York, 1985.

[VER.71] C. BERGREN. "*A simple Algorithm for Circular Interpolation*", Control Engineering ,September 1971,pp. 57-59

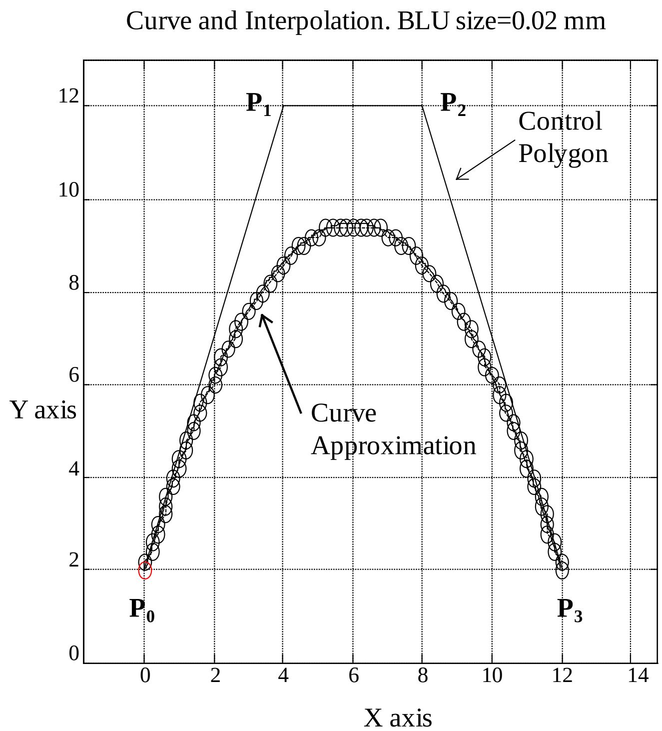

Fig.7

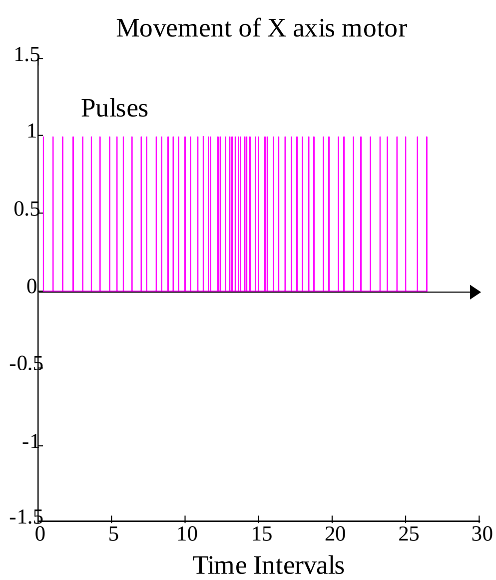

Fig 8. Secuence of pulses X asis motor for curve in Fig. 7

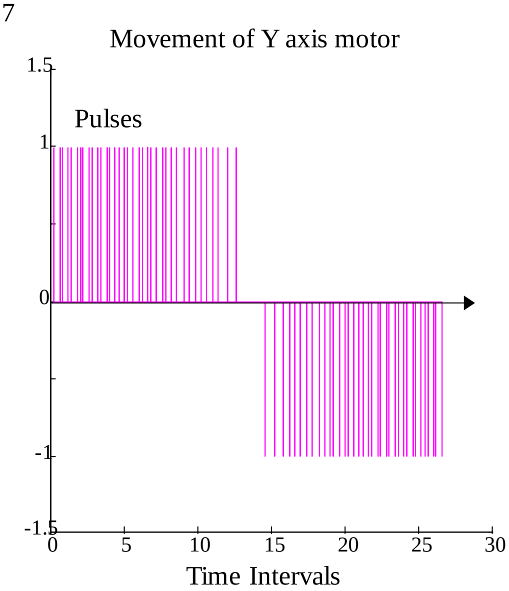

Fig 9. Secuence of pulses Y asis motor for curve in Fig. 7