Using CAD to Help Write G-code

by Pete Sorenson

Photos and drawings by Author

Some of the reasons to hand write G- and M-code include developing a better understanding of computer generated code, as well as creating short, easily edited toolpath code. The example part shown here requires 14 lines of code, whereas the same part using a well-known CAM program generated 88 lines of code.

This discussion will focus on what at first glance appears to be a simple 2D part. However, it's not so simple. The level of difficulty is due to tangencies between the lines and the arcs. If the part was a simple rectangle or arc, the CAD drawing could be unnecessary, but as we want to write the toolpath for the center of the cutter, not the edge of the part, there is a large quantity of mathematics involved with writing a toolpath for this part. I'll explain how I use CAD to help with all that math.

Figure 1

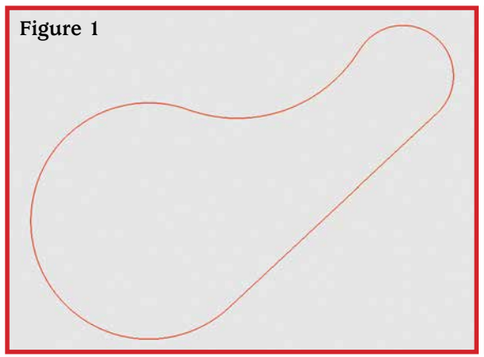

Figure 2

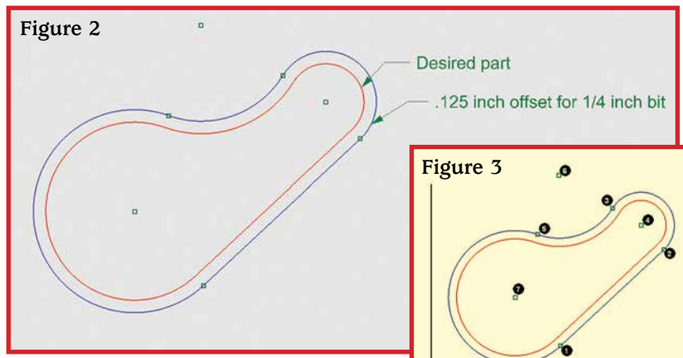

Figure 3

First, draw the part using your CAD product of choice; I use Rhinoceros V5. Place your drawing in the same orientation and relationship to X and Y Zero as your desired part will be when on the milling machine. Use your CAD product's built-in analysis tools to make sure there are coincident points where the lines and arcs intersect. At this point, you will want your drawing to be free of extraneous information and overlapping lines and arcs (Figure 1).

Next, offset the lines and arcs by the radius of the bit you are going to use (Figure 2). Depending on the precision desired, either accept the nominal radius of the bit or use an offset that is half of the measured diameter. If you want a finish pass, you will have to do two offsets. The first offset would be the radius of the bit plus the amount left for the finish pass and the second offset would be the radius of the bit.

I have numbered the points that designate the ends of lines and the arc centers. The numbers are optional, but I find them helpful as I "write my way around the part" (Figure 3).

Now, use the dimensioning or analyze tools in your software to assign an X and Y value to all the necessary points in the drawing. This will be wherever a line changes direction or where an arc starts or ends, as well as the centers of the arcs. The example is dimensioned using ordinate dimensions (Figure 4).

Use at least four place dimension and drawing precision. Even if your part does not need the tolerance, it will be required by the controller software to produce the desired arcs.

You now have the information needed to write your code. Your first decision will be choosing between conventional and climb milling, as this will dictate the direction around the part. The example moves in a counterclockwise direction, producing a conventional milling toolpath, as would be recommended for a lightweight or tired machine. I am using a Light Machines ProLight 1000 and have found it does not like to climb cut.

On some lines I have placed a semicolon. The semicolon and information following it are invisible to my controller. Most controllers have some means of inserting comments into the G-code. Use this to put in operator information, especially if someone else is going to run the machine or if it will be awhile before you use the toolpath. The code for the sample part is shown in Listing 1.

If available, use your verify feature to try out your program or set the Z zero high enough that you can air-cut the part before you actually cut material. I will often machine a test piece from Melamine-covered particle board, mainly because I have plenty of it in my shop scrap bin.

There are a few things you will need to know about your personal control software. The toolpath shown works with my machine but might not work with yours. Some controllers require line numbers, either as N, N2 or N01, N02. Other controllers need a 0 before a decimal point for values of less than 1. Your controller might require a 0 in the G- and M-codes that are below 10; for example the codes would be entered as G02 or M03, rather than G2 and M3.

Arc centers can be either relative to the start of the arc or expressed as absolute X and Y values. This may be an operator choice on some machines.

Another place that may take some work is programing Text. If you draw the Numbers or Letters yourself and make sure your line endpoints are coincident where they meet there should be no problems. If you use a text not made for machining there can be overlapping line segments, as well as very short segments that make it difficult to obtain the correct X-Y values.

Photo 1 shows an example part made with this method of creating code. Because of the small bit diameter and the thickness of the plate, it required several cutting passes with increasing depth. Using the Copy and Paste function in Notepad, it is easy to make multi-pass code. Simply edit your code, increasing the Z-value for each copy to reach the total depth desired.

While this is just one of the many ways to go about creating G-code for your machine, I think you will find it is a quick and easy method.

Listing 1

```
G90G17; Sets absolute coordinates and X-Y plane
; Use .250 bit
; Top of part is Z0.0
G0Z.250; ensures the cutter is clear of the part
G0X1.9655Y.5368; Places cutting tool over point 1
M03S1500; Turns on spindle at 1500RPM
G1Z-.125F4; Moves cutter to -.125 at 4 inches per minute
G1X3.5364Y2.0002F10; Cuts to point 2 at 10 inches per minute
G3X2.7602Y2.6345I3.1916J2.3703; I and J are the X and Y values of point 4,
; The X and Y are point 3
G2X1.6173Y2.2325I1.9417J3.1358; Arc ends at point 5. Point 6 is center
G3X1.9655Y.5368I1.2746J1.2785; returns to point 1
G0Z.250; lifts cutter clear of work
G0X0Y0
M05; turns off spindle
M02; ends program
```

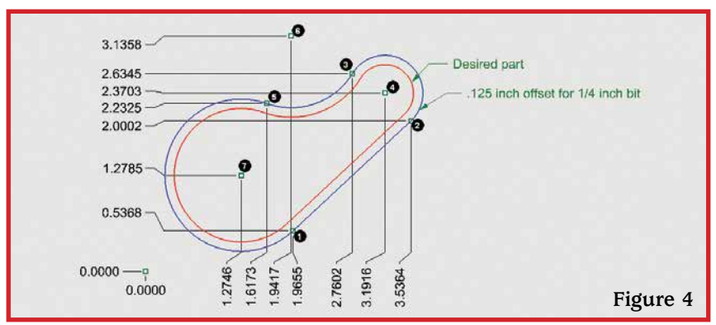

Figure 4