Here is what you need to do to get **WarpDriver** installed and running on your Windows-based computer. Current Version is 2014-01-18.

Warning

These instructions are way too long.

Disclaimer

By downloading, installing and using this program you are accepting full responsibility for any and all consequences. CNC machinery is potentially dangerous, and the user is 100% responsible for ensuring that the output of **WarpDriver** is safe to use on any CNC equipment, and that it will have the desired effects.

My Aspire, VCarvePro and PhotoVCarve programs only use G00, G01, G02 and G03 movement commands in the Mach3 postProcessor, and these are all handled correctly by **WarpDriver**. My testing has been exclusively with the Mach3 inches PP. G-code generated by other CAD/CAM programs or other PostProcessors may include other G-codes, and ***you*** must verify that the output is coherent and viable.

As always with CNC equipment, think many times before running code, and doing air cuts is often a good idea with new code generators. In Mach3, verify the Z limits of the loaded files before cutting to ensure that you are not going to destroy your table top, or spindle, or both. The XY limits should be identical to those of the original Source G-code file unless using the Scale then Shift model, but the warped Z can be much deeper, depending on the model. **Play safely**.

What Does WarpDriver Do ?

Most 2D CAD/CAM programs prepare cutting profiles that are relative to a flat surface of the work, usually with Z=0.0 as the reference plane. You control the depth of cut and the outlines, but this reference plane remains constant across the entire work piece. This can produce terrific designs and wonderful projects. **WarpDriver** extends this by allowing you to define reference surfaces that are more complicated than simple planes, and 'warps' the toolpaths to be measured relative to these surfaces. If a cut was originally designed to be 0.1" deep at some point below the Z=0.0 reference plane, (*x,y,-0.1*), and we define a new reference surface that is 0.5" below the Z=0.0 plane at these (*x,y*) coordinates, then the warped toolpath would pass through (*x,y*, -0.6"). If the original toolpath was a simple 0.1" deep groove going from left to right, and the warping model presented a dished surface, then the warped toolpath would be 0.1" deep outside the dish, then follow the dish down into the work (always 0.1" below the dish surface) and then rise back up to a simple 0.1" deep groove on the other side; the groove would be visible when looking down on the work piece. Although dishes can be programed in VCP using flutes, it is not possible to use the fluted dish as the cutting surface for subsequent cutting operations. The program goes beyond just translating z values. The CAD/CAM programs create paths assuming that the reference plane is regular, and flat; when we change this assumption to use a structured reference surface we must remap all of the Z coordinates at every point between the start and end of every move. You can control the accuracy of this interpolation from the front panel of **WarpDriver**. At the present time, the moves that **WarpDriver** understands and reinterprets include G00, G01, G02 and G03 modes (rapids, machining moves, CW and CCW arcs, respectively). Arcs are transformed into a series of G01 moves with closely spaced points around the arc. **WarpDriver assumes that the G-code and machine interface are using the incremental (I,J) arc notation (which is more accurate than the absolute mode).** These seem to be the only moves that Vectric's VCP and PVC programs generate, at least for the Mach3 PostProcessors that I use. **WarpDriver** ignores all other G codes, and moves them to the output file without modification. ***Your mileage may vary.***

Because **WarpDriver** operates on the toolpaths generated by the CAD/CAM programs, *any* G-code they produce should be warp-able. This would include images from PhotoVCarve, text in fancy fonts, V-Carved, prisms, … just about anything.

WarpDriver is not a substitute for a full 3D cutting program such as Vectric's Aspire, but it can extend some of the capabilities of the 2D programs like VCarvePro and PhotoVCarve. Unfortunately, **WarpDriver** knows nothing about the shape of the bits that are used to cut the surfaces, and so the final reference shape cut into the material may differ slightly from the mathematical model. This problem will be observed mostly at the sides of steep structures and when using wide bits, but then, you would not plan to use broad bits for doing 3D machining. It should not even be visible for most artistic or sign cutting with reasonably sized, ball-nosed bits. Imposing PhotoVCarve toolpaths (or any others that are running very shallow cuts) onto curved surfaces can lead to a loss of detail near more highly sloped surfaces.

CSV Output

Although you can use the warped gcode directly on your CNC hardware, there are certain risks with this; previewing is always a good idea before pressing the 'Run' button. One way to preview is to use a program like NCSim or KMotionCNC. Another is to reload the toolpath back into the CAD/CAM software. **WarpDriver** can prepare files into comma-separated-values (CSV) format of (X,Y,Z) toolpath coordinates that define the tip-of-the-tool motion, which can be read back into Vectric's Aspire and V-Carve Pro programs using the CSV gadget. All other information is stripped away. Within the gadget you identify the tool to be used, and the Vectric codes will make the customary previews as if you had used their internal drawing and toolpath generation methods. From here you can continue to work around the imported toolpaths, or output gcode as normal.

Important Note : The CSV Gadget interprets the semicolon (";") as separating vectors in the file, and it will lift the tool between vectors. It is important that you turn OFF the 'Verbose' option when making production CSV files, or else your read in toolpath will preview like a vertical forest of tool raises, and the cut time will go way, way long.

WarpDriver's CSV Output option also lets you convert anyone's gcode into CSV data for use with Vectric codes; just don't check any code segments before creating the output. Please carefully consider all copyright issues before using this capability; gcode produced by others (and CSV files based on this gcode) is probably protected intellectual property, and should not be distributed without the express permission of the gcode author.

Skew Correction

As of the 20120102 version of **WarpDriver**, it can remove residual non-orthogonality from the X-Y axes of the CNC machine using a parameter called the X-Y Skew Angle, located near the right side of the control window. By default, this value is 0.00, and no skewing correction is applied. On my machine, looking down on the X-Y axes, the Y+ direction is only 89.882 degrees from the X+ direction, instead of 90.000. I enter +0.118 as the X-Y Skew correction, and ALL g-code moves will adjust the X values to mitigate these imperfections. Y coordinates are unaffected (the effect on Y is VERY small for small angular errors). You can type this value in every time you run WarpDriver, or you can provide the following on the command line used to execute the program (probably via a shortcut)

```
WarpDriver.exe     skew=0.118
```

One downside of this 'feature' is that a X-Y Skew will (must) convert an arc to a set of steps, even if that toolpath is not be Warped in the Z direction. Files can get very large. If you are not using skew corrections, and if you use arcs on unwarped toolpaths, and you find the files are too big, consider saving these toolpaths to a separate file that don't get processed.

Some Caveats

In addition to the complete lack of warranties and assurances that this program will do what you want safely, there are some minor restrictions that you should be aware of.

1) This program has been tested on the G-code produced by Vectric's V-Carve Pro and Photo V-carve, and MecSoft's FreeMill, using the Mach3 Arcs (inch) preprocessors. No problems were encountered. I don't know of any reason that other preprocessors would cause difficulties, but that is completely unverified. I cannot make any comments about G-code generated by other programs. The program can deal with G00 [rapid move], G01 [machine move], G02 [CW arc]and G03 [CCW arc] movement codes well, and should simply passes all others through to the output with no modification. No new control codes are added, although G02 and G03 are converted into G01's.
2) The warping models provided with the program all assume that Z=0.0 is the topmost surface of the material. If you define Z=0.0 to be the bottom of the material, simple warping should work with no problem (but please verify the G-code with air-cuts and a G-code viewing program). **However, roughing passes using the Passes parameter are 100% guaranteed to fail, and in doing so they will plunge your tools deeply into the material on the first pass.**
3) For best operation you should add some simple comments into your original G-code. This could be done manually in **WarpDriver**, but it is better to modify your PostProcessor to automatically do this every time you prepare some G-code. The comments have no effect on the execution of the G-code, original or warped. Modifying the PostProcessor files is easy, but you are responsible for ensuring that this has been done correctly; it is a good idea to make copies of the PostProcessor file first, 'just in case…'. Again, verify the original and warped G-code before cutting. Instructions for the changes need are included below, along with examples for the Vectric PostProcessors that I have worked with.

Installation Instructions

1) Download the ZIP file, and decompress it into the directory of your choice, using the decompression software of your choice.
2) Run a virus checker on the extracted programs to ensure a clean transfer. If a problem is found, erase the downloaded files; if you believe that the file was infected at the source, please contact me through the PaulRowntree.weebly.com website.
3) Make sure that the directory **c:\temp** exists on this machine. **WarpDriver** uses this directory for temporary files (**temp.ngc**), that can be safely erased at any time.
4) There is no Windows registration process. You can move the **WarpDriver** directory to any location or file system that you wish. You can run as many copies of this program, on as many computers, as you wish. You may also make any donation that you wish to support this program's development through the PaulRowntree.weebly.com website.
5) While not necessary, **WarpDriver** is easier to use if you have a G-Code viewer program available. I have used Dynomotion's KMotionCNC program for this, although it has some oddities in this role that requires editing the gcode before the previews. This free program can be downloaded from http://dynomotion.com/Software/Download.html. Follow the appropriate installation instructions. **A better viewer (I think) is Gershon Elber's NCSim, and this Instruction set mostly assumes you are using NCSim**. NCSim can be downloaded from http://www.cs.technion.ac.il/~gershon/NCSim/
6) Optionally, create desktop shortcuts that point to the **WarpDriver** and your previewer of choice executable programs. If you want, include the X-Y skew correction on the command line to **WarpDriver**.

De-Installation Instructions

Delete the files that were downloaded and shortcuts you created. There is no Windows Registry information that must be removed or cleaned up after.

Modifying the Post Processor

Some of the parts of **WarpDriver** need to be able to identify the sub-toolpath segment 'chunks' within a G-code file. The chunks correspond to the individual sections that you wish to process separately (i.e., warping some but not all) or repeatedly (i.e., roughing passes). It is usually not practical to hunt through the G-code to identify these chunks, so **WarpDriver** tries to do that based on the comments in the code. Comments are lines containing parentheses like '**(some text here)**', not including the quotation marks. There may be a preceding line number. The Vectric Post Processors that I have used are very good at adding comments before each chunk, except that the default footer section of my Mach3 Post Processor does not. It is easy to add this to the Vectric programs by editing the Post Processor file, but please read the Vectric instructions so you understand what you are doing; for other software you must learn how to do this if you wish to use the automatic chunk detection mechanism of **WarpDriver**. If you cannot ensure that the chunks are properly identified (see below on how to decide), do not use the roughing or selective warping options of **WarpDriver**. Warping does not change comments; they are passed verbatim on to the output to help document the generated G-code.

A further (optional) change is to add safe-start commands to the start of the segment chunks to ensure that the tool is clear of the material before starting. This is only important for using roughing passes, where the same source code is warped several times, and must be able to start in a known, safe configuration.

V-Carve Pro 6.5

1) In VCP, go to the File→Application Folder. A window will open with several folders; open the PostP folder and find the one you want to modify. I suggest you make a copy with a different name, and manipulate this file instead of the original. Don't forget to select the modified PostProcessor in VCP!
2) Open the file in the text editor of your choice; NotePad is a good choice, word processors usually are not.
3) Near the bottom of the file, make the changes shown here. **Add the lines shown in BOLD font.**

```
+---------------------------------------------------
+  Commands output for a new segment - toolpath
+  with same toolnumber but maybe different feedrates
+---------------------------------------------------

begin NEW_SEGMENT

"[N](Toolpath:- [TOOLPATH_NAME])"
"[N]([TOOLPATH_NOTES])"
"[N][S]M03"
"[N]G00[ZH]"
"[N]G00[XH][YH]"

+---------------------------------------------------
+  Commands output at the end of the file
+---------------------------------------------------

begin FOOTER
"[N](File footer)"
"[N]G00[ZH]"
"[N]G00[XH][YH]"
"[N]M09"
"[N]M30"
%
```

PhotoVCarve

The G-code files produced by PVC are very simple, although they do not have many comments in them to delimit the segment chunks. This is usually not a problem because (a) there is only one toolpath in a file, and (b) you never need to do roughing runs because the model shape will have already been exposed in a previous warped pocketing operation and the lines are very shallow.

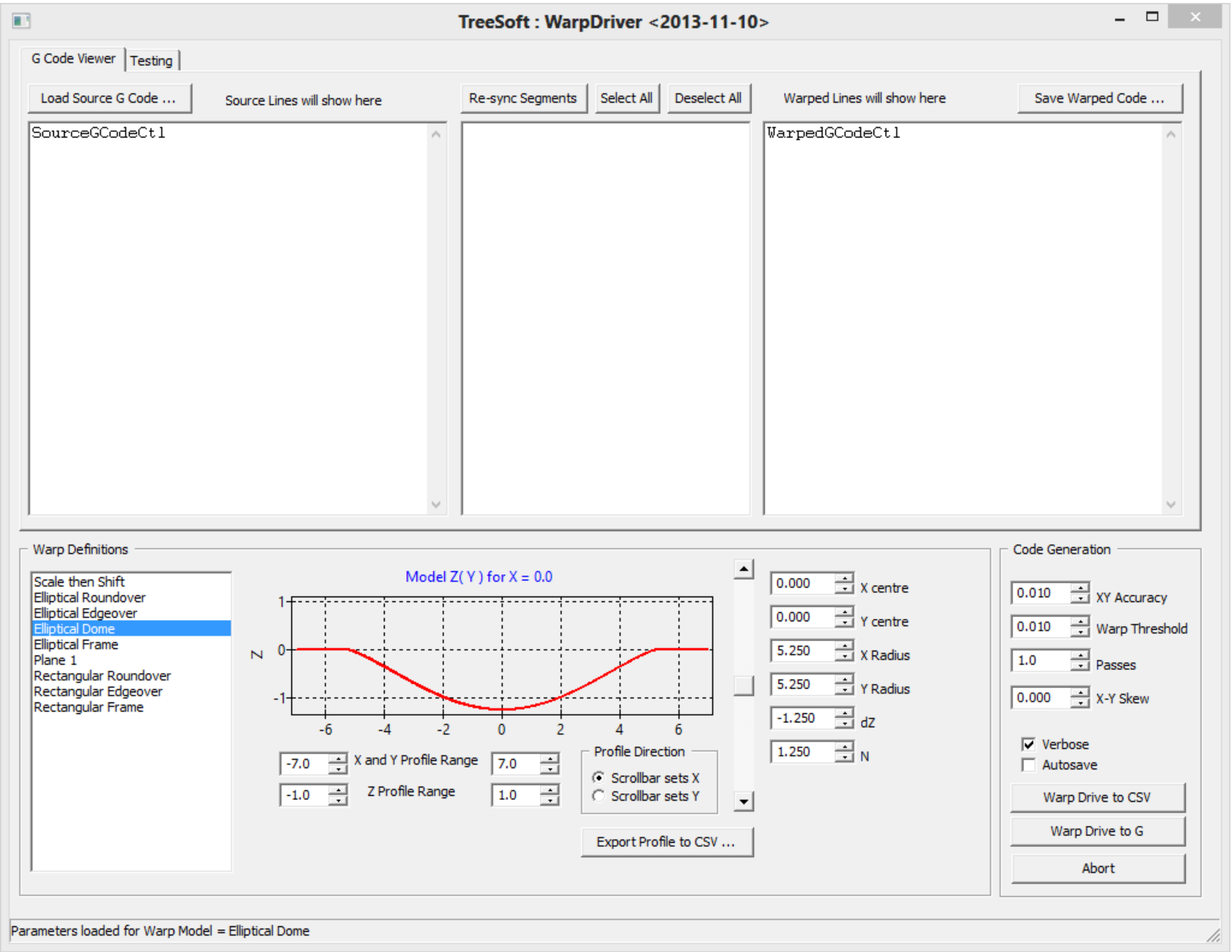
A Quick Run Through the Program

1) **Prepare some toolpaths to Warp**

   There is a Vectric V-CarvePro file called ThinPockets.crv included in the directory. This file contains several rectangular and circular areas. Adjust the stepover and tool as you see fit; ball-nosed bits with 0.100" stepover is fine for on-screen work : real cutting will be better if finer stepovers are used. Sometimes it is hard to visualize the G-code toolpaths using the on-screen viewers if the line density is too high. One at a time, select the vectors that may be of interest, and generate pocket toolpaths to a depth of 0.000". Choose useful names because we will want to find them in the G-code toolpath later. Save these zero-depth pocket toolpaths into filenames of your choosing, using the PostProcessor appropriate to your machine. For now, it makes sense to have each of these pocket toolpaths in its own file because in this case you will use them individually, as will be seen below. You *can* have multiple toolpaths processed in one file, and you can select which ones will be warped (e.g. a design and a dado cut that share the same tool, or multiple text carvings). Here we want them one-at-a-time because you will only impose one pocket at a time on your warping model. There is another Vectric file called Engage.crv that includes two more 0.000" deep pockets and a simple text message. Prepare the pocket toolpaths and the V-carved text toolpath to separate files, using bits and parameters appropriate to your machine.

2) **Execute WarpDriver**

   Run the **WarpDriver** executable, and wait for the nagscreen to pass. You should be seeing something like this:



TreeSoft : WarpDriver <2013-11-10>

Scale then Shift
Elliptical Roundover
Elliptical Edgeover
Elliptical Dome
Elliptical Frame
Plane 1
Rectangular Roundover
Rectangular Edgeover
Rectangular Frame

X and Y Profile Range: -7.0 to 7.0
Z Profile Range: -1.0 to 1.0

● Scrollbar sets X
○ Scrollbar sets Y

X centre: 0.000
Y centre: 0.000
X Radius: 5.250
Y Radius: 5.250
dZ: -1.250
N: 1.250

XY Accuracy: 0.010
Warp Threshold: 0.010
Passes: 1.0
X-Y Skew: 0.000

☑ Verbose
□ Autosave

Parameters loaded for Warp Model = Elliptical Dome

3) **Execute the G-Code Viewer**

If you are using a G-Code viewer run it now.

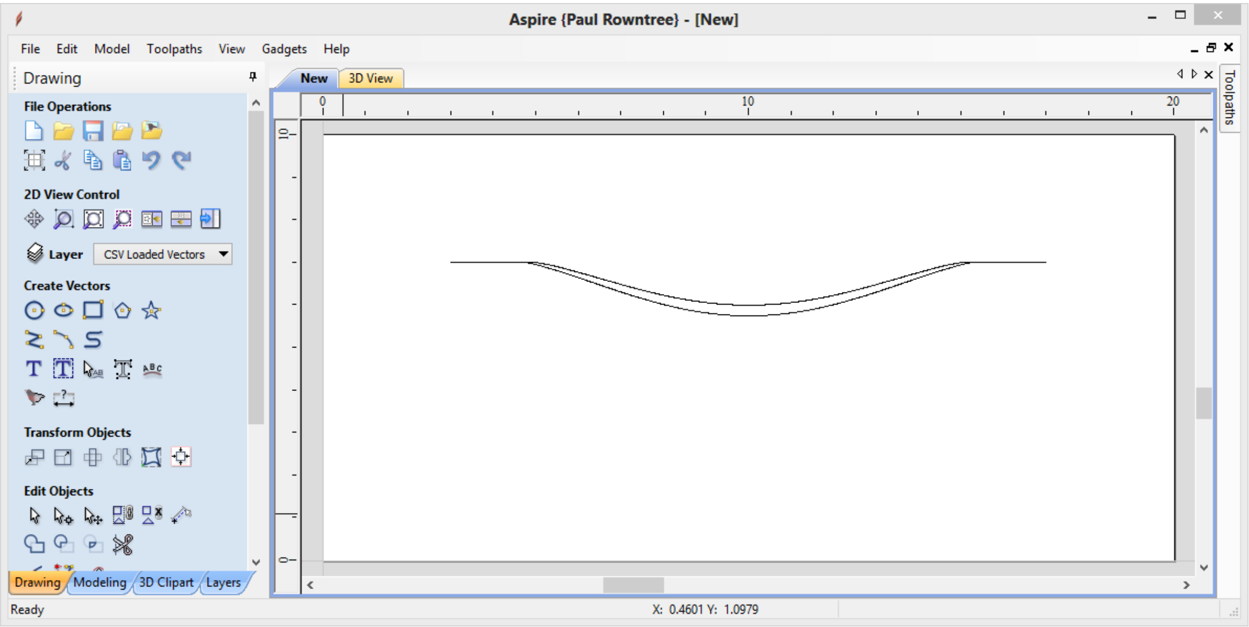
4) **Explore the Interface**

For now we are just going to look at some of the built-in models used for warping. In the bottom left corner there is a list … select 'Elliptical Dome', then click in the graph window to the right. A plot will appear that shows the warping model depth (below Z=0) as you move across in X or Y. If you slide the scrollbar, the graph will change to show different profiles, at different values of Y or X, respectively. You can reverse the choice of profile direction with the 'radio buttons' marked as 'Profile Direction', that tell you what profile is being shown on the plot. Either way, the graph shows you a cut through the model for a constant value of X or Y. The number entry boxes at the bottom set the limits to the graph. The graph refreshes itself when you click on the graph, move the scrollbar, change a graph limit or change a model parameter. Since the Z scale is usually much smaller than the X or Y dimension, the graph may give a distorted impression of how much warping is being done. If in doubt, make the scales similar.

5) **<u>Explore the Parameters</u>**
The list of number entry boxes to the right of the graph show the parameters used in this model, and their names. Parameters are specific to the model; when you change models, the values are saved, and retrieved when you come back to that model; they are reset at run time to default values, so write down important values. You can fool around with these and see how they work. If you give a negative 'height' parameter, the dome turns into a dish. The two radii set the size of the elliptical envelope, and the X centre, Y centre give the centre of the ellipse. For now, keep them at (0,0) because most of the pockets that you created from the ThinPockets.crv file are centred on (0,0). Note that the centre of the model in **WarpDriver** does not have to be centred on the design that you want to cut; you can use just the edge of the model if that is what you need to do. The 'N' parameter controls how the shape of the elliptical warping model evolves across the transition region, with values >1.0 giving a more sharply peaked shape with sloping sides, and values <1.0 giving more of a rounded top pillbox profile with steep sides. Each parameter is sanity checked before use, so absurd values won't (shouldn't ?) crash the code, but check to make sure that the model looks the way you want.

6) **<u>Explore the Models</u>**
Browse through the other models, and see what the various parameters do and the profiles you get as you move the scrollbar. '**Elliptical Frame'** is interesting. When you are done playing, go back to the Elliptical Dome model and put in some reasonable values (Radii of 2.0 and 4.0, N=1, Height=1.0). You will also see a button "Export Profile to CSV", which will take the data in the graph and output

it to an ASCII (text) file containing two columns of numbers. This data can be directly loaded into Vectric programs (Aspire version 4 or greater, V-Carve Pro version 7 or greater) using the TreeSoft CSV gadget package. This data will appear as a 2D vector in these programs. The following screenshot shows two similar profiles that were exported during the development of concave/convex profiles to make a bowl using 2 sided CNC machining. Once the bottom surface of the bowl is defined, you can change the vertical scale to be negative, and shift it vertically as required to machine down the sides to the lip of the bowl.



7) **Load a Source G-Code File**

Now click the 'Load Source G-Code' button, at the top of the left hand text window. Navigate to the ThinPocket G-Code files that you created, and select the one that holds the 6" diameter 0.000" deep pocket. The contents of this file should appear in the window, and a list of some lines appear in the middle Segment List control box. The Segment List shows all lines that include ( ) and they separate the various toolpaths in the file. I have modified my PostProcessor for VCP to make these lines more obvious but this is not required. The numbers preceding the G-Code information are the line numbers in the source file. If you double click on one of these lines, the associated text in the Source G-Code box to the left should highlight this text block; you may have to scroll to find them, but they are there. To ensure that **WarpDriver** has correctly identified the segment 'chunks', look at the selected code lines and see if they correspond to the natural expected divisions. In particular, make sure that the end of the last code chunk does not include the file footer lines, which might shut off the spindle, and terminate execution; these would obviously be bad in the case of roughing path generation. If **WarpDriver** did not find the end of this last segment, and you want to use selective warping of parts of the file or roughing passes, then you must insert the comments to delineate the chunks yourself (or modify the PostProcessor to do this for you), and then **Resync the segment list**.

8) **Select the Toolpath(s) to Warp**

Click the 6" Spiral toolpath checkbox. The checks tells **WarpDriver** which toolpaths are to be warped; all segments that are not checked are copied to the destination without warping, renumbering as required. **Each time you process the G-Code you process the *entire* source file, modifying only the segment chunks that are checked.** Since this file only has a single toolpath in it, you could also press the 'Select All' button which is above the Segment list. Comments from the original file are included in the output file even if the segment is not checked.

> **KMotionCNC Users** : If you just want to see the toolpath, then don't check any segments, and continue on. It is best to use KMotionCNC with the temp file produced by the warping operation (C:\temp\temp.ngc) instead of your original G-code file, because KMotionCNC seems to modify the G-code file sometimes, and in general we don't want this.

9) Prepare for WarpDrive …

On the right hand side there are two check boxes, 'Verbose' and 'Autosave'. 'Verbose' tells **WarpDriver** to include lots of comments in the generated G-Code, while 'Autosave' tells the program to save the results to the last-used destination filename. This defaults to the loaded filename, with 'Warped ' preceding the filename, but this can be changed by saving to a different filename. **Turn off the Verbose option when using CSV files.**

Three other controls should be considered, on the right side above the Warp Drive button.

XY Accuracy sets how precisely to interpolate the toolpath in creating the output; smaller values give better tracking, and larger output files. In my work so far I have used the default value of 0.01" and it seems fine. If you use mm units, then an Accuracy of 0.250 would be more appropriate. **WarpDriver** doesn't have any units preferences internally, but the default values are currently more appropriate for inch units.

Warp Threshold is a parameter that allows the code to ignore modifying the Z coordinates of lines when the original and warped toolpaths are always above the specified value (and therefore not in danger of running into anything). Since Z=0.0 is often the top of the work piece, this default is a reasonable threshold value. Note that some models may warp the X and/or Y axes (e.g., "Scale then Shift" model); the X and Y warping takes place for all segments checked, regardless of Z being above or below Warp Threshold.

Passes is a parameter that allows roughing passes to be created with the current toolpath and the current model; instead of going directly to the reference surface with a zero depth pocket in a single step, it will work its way down by repeating the pocket the number of times you indicate in this control. The value is interpreted as 'a.b' (i.e., if the value is 4.2, 'a'=4, 'b'=2). Each roughing pass will advance the tool into the work by (Total Depth/a.b), and there will be 'a' of them, followed by a single finishing pass to complete the cut to the full total depth. Passes=4.2 will create a toolpath with 5 passes in total with the final cut that is 0.2/1.0=1/5th the depth of cut of the previous 4 passes. The maximum depth of cut for these passes are then 0.238", 0.476", 0.714", 0.952", and 1.000". Passes>1.0 slows down the machining process, but it is probably necessary for fine radius bits that have short tool lengths and even shorter cutting surfaces. See the note below on alternative roughing methods. The default value of Passes=1.0 (i.e., the cut is a single pass to full depth). Z motion above the Warp Threshold is not affected by Passes. When imposing patterns *etc.* onto the surface you will probably use Passes=1.0, since roughing control should have been done within the CAD/CAM programs.

10) **Engage!**

Finally, press one of the 'Warp Drive' buttons near the bottom right side of the window. You can create GCode or CSV values according to the button selected. After a few seconds a message box will appear to tell you what the minimum Z value has been encountered in the entire warped G-code. Since the original pocket was 0.000" deep, and our pocket extends across the entire model ***in this case***, the minimum Z should be the height of the model. If the Z Minimum seems too shallow, verify that you have selected the correct segment 'chunks' and that the model parameters are as expected. If you make the model's X and Y radii larger, the 6" diameter pocket will cover less of the model, and in this case it will lead to a minimum depth that is between the height of the model and 0.0 . The Destination Code box should fill up with the processed G-Code, and if you are fluent in G or CSV you can see what **WarpDriver** has done.

11) **What is the CSV All About ?**

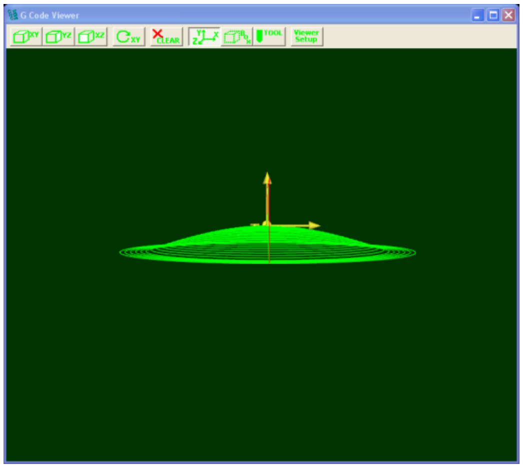
The CSV output is "Comma Separated Values", commonly associated with spreadsheet programs. It is also a format that can be read into Vectric programs (Aspire version 4 or greater, V-Carve Pro version 7 or greater) using the TreeSoft CSV gadget package. These programs can directly load the CSV files created by **WarpDriver** and you can then preview the newly created toolpaths. From there, you can explore it with the Vectric previews, then output it as an official Vectric toolpath. This is very cool, and can relieve a lot of anxiety about alignment etc. Strongly Recommended. **Remember to uncheck 'Verbose' when using CSV output.**

12) **What Does the Created G Code Look Like?**

The rest of us can use a G-code viewer. In previous versions of **WarpDriver** I recommended KMotionCNC (http://www.dynomotion.com/Help/KMotionCNC/KMotionCNC.htm), but I have since found an easier, more useful viewer, NCSim (http://www.cs.technion.ac.il/~gershon/NCSim/); both are free downloads.

NCSim : this program works very well, with fewer issues. Load the **c:\temp\temp.ngc** file, and do the preview. You can also run NCSim using specific tool shapes to get a more accurate view of the output toolpath effects.

KMotionCNC Users Beware: Switch to KMotionCNC, press 'Open', navigate to and select the **c:\temp\temp.ngc** file. The warped G-code will appear in the instructions window. **Here is the oddity** : KMotionCNC doesn't understand the command G70 or G71, which is found near the top of the file in Vectric's header (at least for the Mach3 PostProcessor). G70/G71 sets the units to inches or mm; KMotionCNC apparently uses the equivalent G20/G21 commands. On my PhotoVCarve output the G70 is on a line by itself; on VCarvePro it is with the all-important line that also establishes the arc-definition mechanism G91.1 (incremental I,J); Mach3 needs these instructions! You can safely erase this line altogether in KMotionCNC, and then press the 'Execute' button. Another oddity is that KMotion seems to save the edited file without asking your permission; **for this reason, it is not a good idea to view/edit the original source G-code or the 'Saved' toolpath files with KMotionCNC; only view the 'c:\temp\temp.ngc' file, which is only used as a transient file and by KMotionCNC.** KMotionCNC will then 'run' the G-code, which should take a few seconds. On the right side of KMotionCNC's button bar is one called 'G-Viewer'. Press it, and a new window will appear showing a 3D view of the G-code that you have created. If this window is open, then reloading and re-executing G-code will update the viewer automatically.



13) **Walk Around the Warped Toolpath**

Recall that the original pocket was 0.000" deep. The Warped G-code shows a domed output, as per the parameters used. You can move the 3D view around using the mouse and left button to shift, right button to zoom, and both to twirl.

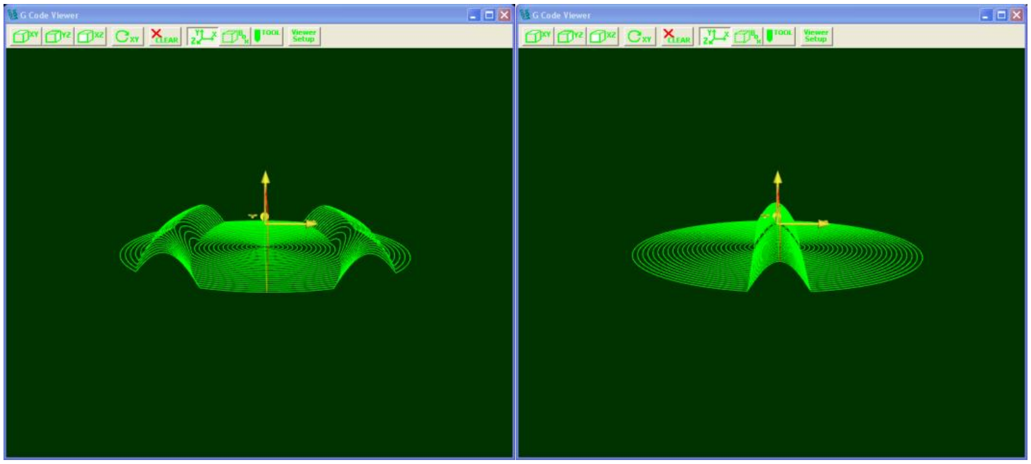
14) **Repeat As Necessary**

Go back to **WarpDriver** and play with different parameters and different models. The temp.ngc file is recreated whenever you press 'Warp Drive', and this is easily re-opened, re-executed, and the display refreshed in KMotionCNC. You will have to keep erasing the G70 line each time, OR you can erase it from the source file (or put it in (comments)). **Remember to reinstate this line before making code that will actually be run on real hardware**.

IMPORTANT : whenever you modify the text in the Source G-code text box in **WarpDriver**, you must **(MUST!)** press 'Re-Sync Segments' (right above the list box). This resets the line numbers associated with each code block; the warp processor uses this to decide which lines are to be warped, and which to more-or-less ignore. It is easy to forget this, and there is nothing to tell you that an error has been made (unless you want to wade through the output, line-by-line).

When in doubt, resync the segments.

Here is the same 6" spiral pocket being used on the Elliptical Frame model. On the left image, the Y radius has been deliberately set to be larger than the radius of the pocket, just to show what can be done. If the Y radius was set to be enormously large, it would give the effect of two parallel ridges; offset the model in X with respect to the pocket and you can have a single long raised ridge of controlled shape, rising above the new background, onto which you are able to carve anything you want. Similar effects could be created with the Elliptical Dome model, which has only a single 'bump' and does not require the offset in X and Y. There is a lot of room for experimentation here. The on-screen profile graph can make tweaking the parameters a bit easier.



15) **Enough Reading, Time to Do Something**

These long-winded instructions should be enough to get you started. Remember that the warping process bends the final toolpath to follow the defined model. If you want to cut the model surface, just create a warped toolpath to the shape you want. If you want to have text or artwork cut into this model surface shape, you must first run a warped pocket to expose the model surface, then run the warped artwork toolpath to get the final cuts. If you neglect to run the warped pocket first you will be trying to do some very deep carving with a tool that is probably inappropriate, and a feed rate that is probably dangerously wrong. Choose your feeds and speeds of the pocketing operations with caution. The main problem with the **WarpDriver** approach is that we are manipulating toolpaths, not design vectors. **WarpDriver** does not work with any tool dimensions, and so it will tend to overcut convex and concave shapes; smaller ball-nosed tools will minimize the overcut. Furthermore, we don't have Vectric's wonderful preview images to guide us unless you use the CSV output option, and we must instead use our wits to keep us from damaging our machines and projects. Find a G-code viewer that you like and make sure to fly in and around your G-code before running it on your hardware.

You should be ready to try the Engage.crv file now. Make the pocket and text toolpaths in VCP. View the original, unwarped toolpaths using NCSim or KMotionCNC by not checking any of the Segments but do not change the G-code within KMotionCNC or it may save these changes without warning you or asking your permission. NCSim has no such issues. The pocket path should be in the Z=0.000” plane; the text path should be normal, shallow cuts, arranged just below the Z=0.0 plane. Warp the pocket with your choice of parameters; I used the Elliptical dome model with the two radii being equal to the radius of the artwork. Repeat the warping with the text toolpath selected (same model and parameters!), and you can see how the corners of the V-carving toolpath are bent down to follow the dome shape. If you are confident that all is well, try some cuts.

Summary

Once again : In most cases, the sequence will be:
1) Execute the warped pocket to expose the desired model surface, then
2) Execute the warped text or artwork path(s) onto the exposed surface.

You should choose your pocket feeds knowing that it could be hogging out to a more substantial depth than your CAD/CAM program anticipates.

While it may seem natural to give the model the same dimensions as the design you are cutting, and to have them centred on each other, this is not required, as the example of the Elliptical Frame showed. For example, you could warp only the centre portion of a larger project, or only use a middle region of a large model to exploit the features of the curvature that you like. You control the model and its effect on your toolpaths. Since CAD/CAM programs and **WarpDriver** use exact dimensions to position the tools, it is easy to keep the design and model coordinates in whatever registry you wish to impose.

Tips on Using WarpDriver

Carving onto Warped Surfaces

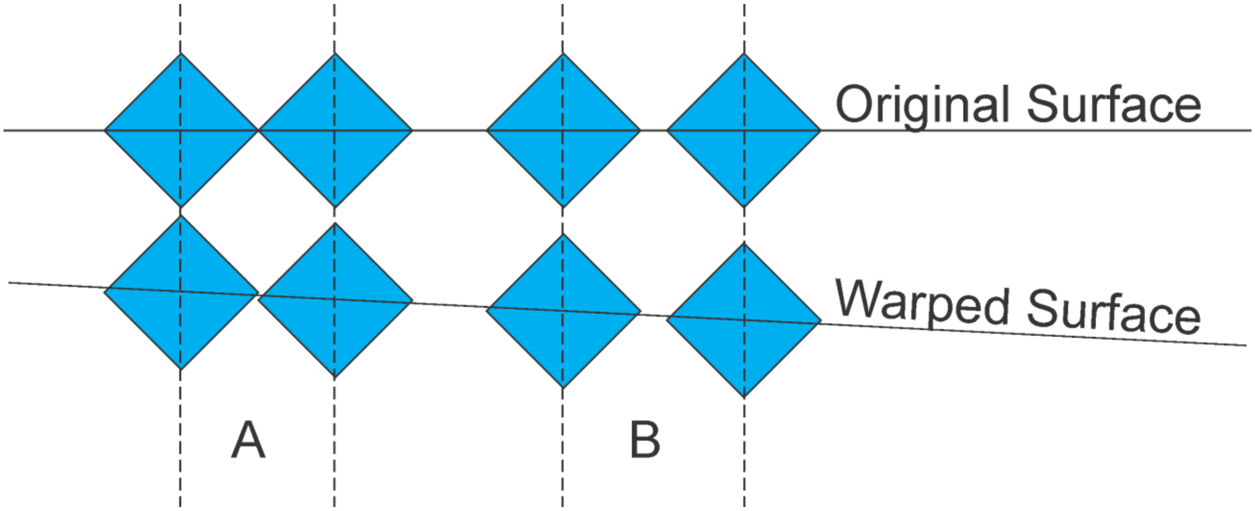
I like the look of putting V-carved patterns onto warped domed surfaces; that is why I wrote the **WarpDriver** program. One potential problem is in choosing the warping parameters for the surface model and for the V-carved pattern. I often use a 0.25” end-mill to expose the domed surface (via the warped Zero-depth pockets), and because the cutter has a width, the final domed shape is a bit *narrower* than the model parameters. If cutting dishes, they will be a bit *wider* than expected. If you use a small ball nose cutter this is less of a problem. You probably won’t notice the difference by eye, but since V-carving is very sensitive to the shape and position of the cutting surface, you would find that a warped carving on a dome would be shallower near the outside edges than in the middle, and this is very visible because the carvings are relatively shallow. To avoid this I expose the domed surface with radii that are ½ of the cutter diameter wider than what I want (dishes would be ½ of the cutter diameter narrower), and then use the true dimensions when warping the V-carving. For example, if the final desired shape is a 10.0"x6.0" domed ellipse to be cut with a 0.25" EM, I would warp the zero-depth pocket with X radius = 5.125", Y radius = 3.125" to expose the dome, and then use X radius=5.00" and Y radius=3.00" for warping the V-carved toolpath. This approximation works well.

Shaping MDF

MDF is a nice material to work with in CNC, if you don't mind the dust. The outside faces of MDF sheets have a harder texture than the inner regions (a 'case'). A pattern that cuts well into the face may show less detail when cutting into the inner regions because this casing is no longer available to stabilize the surface. This became evident when cutting a warped version of the famous 'Mayan / Aztec Calendar', which had the shape of a dome that cut the thickness down to 0.25" at the periphery of the disk. The detail at the edges was noticeably less than in the middle, and there was some chipping out of fine structure. One **WarpDriver** user solved this problem by exposing the domed surface, then coating the MDF with a couple of coats of shellac before V-carving. The shellac stiffens the fibrous surface substantially.

Changes to Feature Sizes

Sometimes features will change width when warped. This is show below for a 90 degree cutter in blue, where a pair of V-cuts that are designed to just touch on a flat surface will overlap when warped in Z (image A). V-Carvings that are wider apart (B) show the same effect, but it is less pronounced. Depending on the slope of the warped area, the regions between V-carved grooves can actually be below the warped surface, creating a variation on the 3D look. For example, the 20" diameter warped calendar has flowers around the edges that showed interesting textures because of this when using a V60 bit.



Cutting Tapered Holes

If you want to cut a tapered hole (e.g., to hold a candleglass) try using a Elliptical Edgeover model, with a negative depth to make it a dish, and a depth that is slightly greater than the thickness of the workpiece you are working with; obviously you will need a spoilboard. The X and Y radii should be the same, and be half the diameter near the bottom of the cut out region. The width is the size of the tapered region. This could also be done with the fluted toolpaths of VCP, with many radial vectors starting on the outside diameter and ending on the inside diameter. Think very carefully about the use of roughing techniques to gently work your way to the bottom of the hole; you could also define a circular path to bore out the hole (no need to warp this) then warp the edges with the Elliptical Edgeover model.

Cutting Reliefs from FreeMill

FreeMill is a free program from MecSoft that can cut reliefs based on STL files and some others that I don't know about. The price is right, but it lacks roughing path generation, so it is difficult to make detailed cuttings. Try using the **Scale and Shift** model, which just scales and shifts the source toolpath by fixed values in Z; there is no model surface for Scale and Shift. If this is warped with **Passes** greater than 1 it will ease the cutter into the work, and then do the final finish pass at the specified dimensions. FreeMill already knows about the cutter dimensions when it prepared the toolpath**. Roughing passes will not work as expected if the top of the material is not Z=0.0.**

Generate your toolpath in FreeMill and save it. Load the G-code into **WarpDriver**, and you will see that the Segment list has two elements; scroll to the bottom of the file. We need to separate the cutting code from the end motor control codes or our roughing passes will end prematurely. You will see a final G1 command that lifts the tool, then some lines with motor commands. Just ***AFTER*** the G1 Z lift, add a comment like '(end section)'. Then Resync the Code List, and you should see three segments listed. The one in the middle with the most lines is the actual cutting pattern, so check it. Now generate the warped path with **Scale and Shift** and the **Passes** that you wish to use. Keep an eye on the number of output lines generated; it can quickly grow to be quite large with STL files. There is probably a way to add the comment by modifying the MecSoft PostProcessor, but this appears to be more complicated than the Vectric PostPro's.

Zeroing the Z-Axis

If you are using a touchplate on the workpiece to zero your Z axis, remember that you are probably going to be working with a curved surface at some point, and it is more difficult to hold touchplates steady on a curved surface than a flat surface. Dual touchplate methods (such as on Gerry's Mach3 2010 control panel) avoid this problem; I do not use Gerry's panel (yet) because I do not have limit switches installed (yet). Consider zeroing the machine on the non-cut regions of the work, and use this area consistently. If you really wanted to go over the top, machine a small shallow flat area to the side of your project cutting area, and put the touchplate on this area for all subsequent zeroing operations.

Roughing

Roughing passes are supported as described above with the **Passes** parameter; this approach assumes that the roughing tool is the finishing tool, and removes material at a user-controlled rate. **As has been repeatedly warned, it only works if Z=0.0 corresponds to the top of the materials.** If this is not acceptable, there are at least two other 'more manual' solutions.

1) You could define a roughing zero-depth pocket appropriate for a larger, more robust tool, possibly with a larger stepover. Use the warped version of this pocket to make the cut in fewer passes; to avoid messing up the sides and edges, set the

depth of the cutter ~0.1-0.25" above the final model shape at the cutting machine, depending on the tool and the edges of the model. I have used a 0.5" End Mill for this purpose. Then run a final cut, to the correct depth, using a more appropriate finishing tool like a ball nose mill on a properly designed single-pass finishing toolpath. This is probably the fastest approach, but it requires another toolchange and Z-zeroing operation. If your roughing tool has a smaller OD than your finishing tool (*could* happen) then make sure that the edges of the rough cut region extend out beyond the OD of the finishing tool (i.e., make sure the roughing pocket is larger than the finishing pocket). A separate roughing path and tool would be appropriate if your finishing tool must be very fine to allow PhotoVCarve work, since this could be prohibitively slow in a multipass approach.

2) You could make multiple passes through the work by starting with Z defined to be well above the material surface at the CNC, and offsetting the Z of the cutter downward on each pass until the final pass is correctly zeroed for the model. Depending on the shape of the model, the cutter could spend a lot of time initially doing air-cuts. This doesn't seem to have any advantages over the roughing control approach built into **WarpDriver** (except for the choice of Z=0.0 issue), and requires more manual intervention at the CNC.

PhotoVCarve

Exposing the model surface with high accuracy is especially important if you want to use the curved surface for PhotoVCarve or similar cutting operations, where the depth of the cutter must be very precise. Slight tracking errors caused by the tool radii will cause visible problems, especially if the model has abrupt changes in slope in the carving area. Cutting tools with the smallest possible radius should be used to expose the surface; this may require use of a roughing toolpath.

If you are going to impose a photo onto a domed (or dished) shape, consider clipping the image to match the dimensions and shape of the dome (or dish). It looks odd to have a square or rectangular image go beyond the curved portions of elliptical shapes, onto the flat areas. The slight differences in the surface finish across this edge can also cause variations of wood staining at these transition areas; this can add a nice visual framing, but not if it passes through your image and highlights the flat-to-warped transition!

Fluting Toolpaths

The standard models that come with **WarpDriver** are fairly symmetric. If you need to impose a curved convex or concave shape with a complex outline onto the material without needing to carve shapes into this new surface, you may want to use the VCarvePro's fluting toolpaths. In particular, I like the way that Tim Merrill suggested where the flutes are run radially out from the middle area of the domed or dished structure, instead of running around the shape in a spiral; the radial lines give more control over the outline shape and can have higher resolution. As coded by Vectric the flutes start as Z=0.0 then gradually cut to negative Z along the length of the vector, ending up at the '**Flute Depth**'. If you want convex shapes (domed) then try the **Scale then Shift** model. The output of this model is

$$Z_{warped} (X,Y) = Scale*Z_{original}(X,Y) + Shift$$

Use **Scale**= -1.0 to turn the concave fluting paths into convex shapes, but you will need to Z shift them ***down*** by the '**Flute Depth**' that you used in VCP to produce a dome with its maximum at Z=0.0. View the warped toolpaths before cutting.

Good Luck!

I hope that you will like the warped look. The warping process does not know (or care) about the content of the Source G-code files, and so it is possible, with care, to warp shapes, text, PhotoVCarve output, … just about anything.

I welcome your comments.

Cheers!

Paul Rowntree

(a.k.a. **PaulRowntree** on the Vectric and CNCZone forums)

Updates to **WarpDriver** are found at PaulRowntree.weebly.com where additional materials are found, and donations to support its development can be registered (and that disable the nagscreen).

Custom and proprietary warping models can be designed to order.