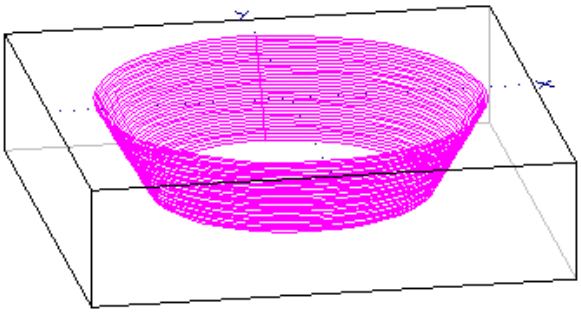
G-ZERO 3D BUILDING BLOCKS MANUAL

Usage of VIEW, +TILT, +TRIM and +PROJECT

Jun 08

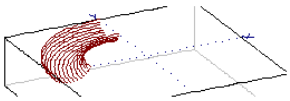
To learn about basic 3D milling:

See pages 2 - 7



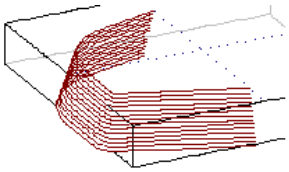
Surface of Revolution

See page 8



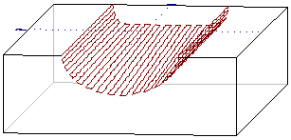
Surface of Translation

See page 9



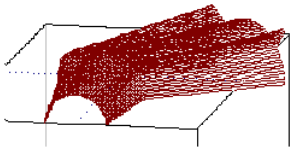
Ruled Surface

See page 10



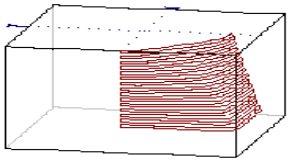
Bent Ruled Surface

See page 11



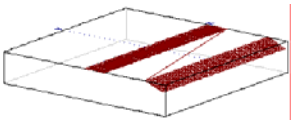
Blended Surface Patch

See page 12



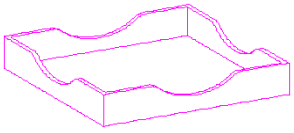
Basic 3D Tilting

See pages 13 - 15



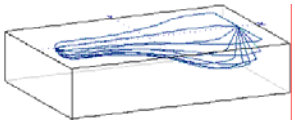
Tilted Contour

See page 15



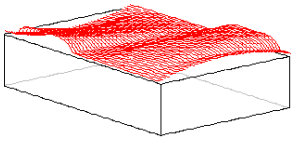
Tilted Surfaces

See page 17



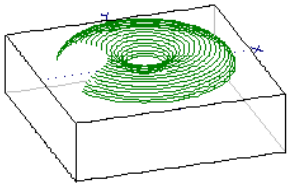
Tilted Patch

See page 18



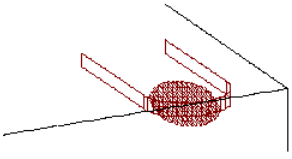
Trimmed Surface

See page 20



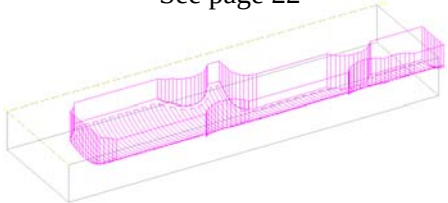
Trimmed Patch

See page 21



XY Projected onto Z

See page 22



+VIEW

Choice 38 in G-ZERO's Mill System menu is called +VIEW (cross view). This command is an optional addition to G-ZERO. To call it up, you may enter 38 directly, or you may click on the menu item.

"Views" Are Input

Before you can use +VIEW to actually create a surface, you must first give G-ZERO some information to work from. This information is grouped into several lines of source code called a "View". A View always begins with exactly one COMMENT which is then followed by geometry like COMP, POINT, RADIUS, LINE, UNCOMP or occasionally STOCK, ROTATE, RECT or ROUND or maybe even ROUGH. Here is an example of a View:

```
6       TV - the next few lines describe my Top View
7 COMP     angle-85 cl/con1 lookahead0
8 POINT    x0 y0
9 LINE     angle275
10 POINT   x.35016 y-4
11 UNCOMP  angle0
```

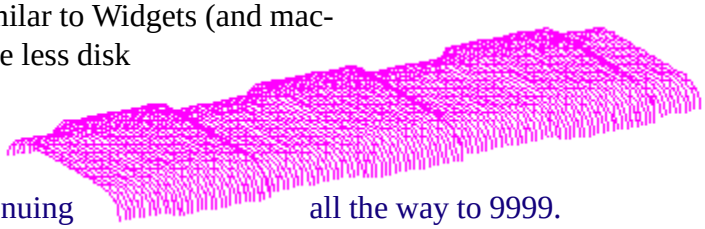
Most surfaces require a couple of Views and so you may string them together by simply following one right after the other:

```
5       Top View
6 COMP     angle270 cl/con1 lookahead0
7 RADIUS   .312 type1 x-.312 y.35455
8 UNCOMP   angle181
9       End View
10 COMP    angle270 cl/con1 lookahead0
11 POINT   x.125 y0
12 POINT   x0 y-.125
13 POINT   x0 y-.5
14 POINT   x.125 y-.625
15 UNCOMP  angle270
```

+VIEW can take up to 4 Views as input.

"Patterns" Are Output

Once it has read in the Views, +VIEW begins to show you the surface. At the same time, all the hundreds (or thousands or perhaps even millions) of surface points are saved in a file on your disk. This file is called a "Pattern". Patterns are similar to Widgets (and macros), but they are compressed and therefore use less disk space and execute much faster. Here is a typical pattern of points:



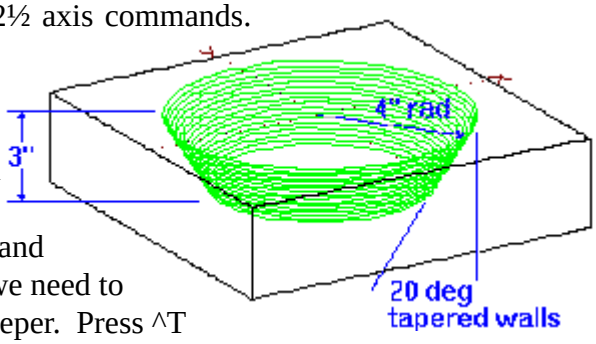
Patterns are numbered, starting at 1 and continuing all the way to 9999.
Patterns are very easily overwritten by other programs.
Unlike other objects in G-ZERO, patterns do not automatically update when a tool diameter or some geometry changes.

In other words, Patterns are powerful and they are fragile. Luckily, it is very easy to regenerate a pattern — just redraw your source with the tool. If you forget this simple rule, your final G-code might not reflect your last edits. Worse yet, your final G-code could contain data from another completely different part.

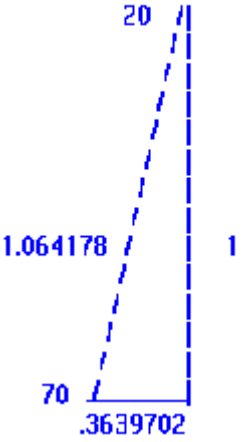
So always be sure to press the F3 key before posting an un-new 3D source program.

In the Beginning

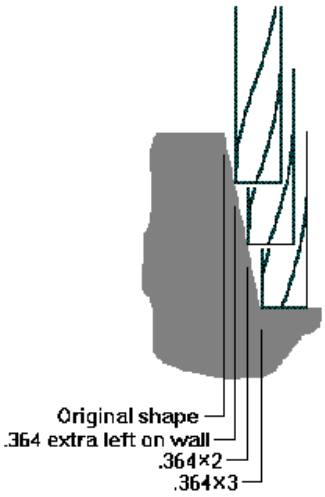
Before we start using +VIEW to create quadzillions of points, we need to take one more brief detour. To get an idea of how +VIEW thinks about Views, it is instructive to create just one simple 3D program <u>without</u> 3D Building Blocks — using only 2½ axis commands.



Imagine trying to rough out the middle of this 8" tapered pocket with a .5" sharp endmill. The first pass will be at Z-1, the next pass will be at Z-2 and the final pass will be at Z-3. Of course, the walls of this pocket are not vertical, so we are going to have to cut smaller and smaller circles as we go deeper. Or, put another way, we need to leave more and more material on the walls as we go deeper. Press ^T RIGHT TRIANGLE (Ctrl-T) and use the triangle routine to figure the taper amount:



For every inch we drop the cutter in Z, we must scoot away from the walls .364 inches.



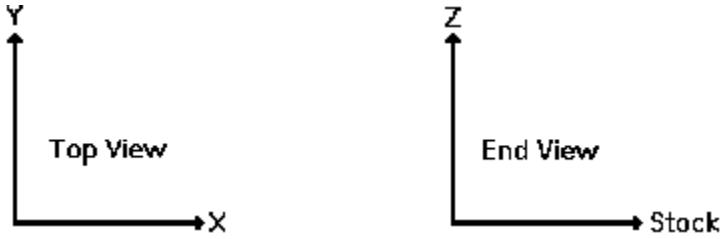
Using the ROUND command along with the STOCK command, this program becomes simple:

```
1 MAT'L    xmin-5 xmax5 ymin-5 ymax5 thk4
2   TOOL 1 dia.5
3 STOCK    .364 zstk0
4 MILL     zrapid.1 zcut-1
5 ROUND    dia-8 x0 y0 thru0
6 STOCK    .728 zstk0
7 MILL     zrapid.1 zcut-2
8 REPEAT   from5 thru5
9 STOCK    1.092 zstk0
10 MILL    zrapid.1 zcut-3
11 REPEAT  from5 thru5
```

If you take the time to type this program in, you may save yourself weeks of frustration trying to remember how “End Views” work. Please take the time right now. The concept of repeating some shape over and over again, but leaving different amounts of STOCK each time is the secret of +VIEW.

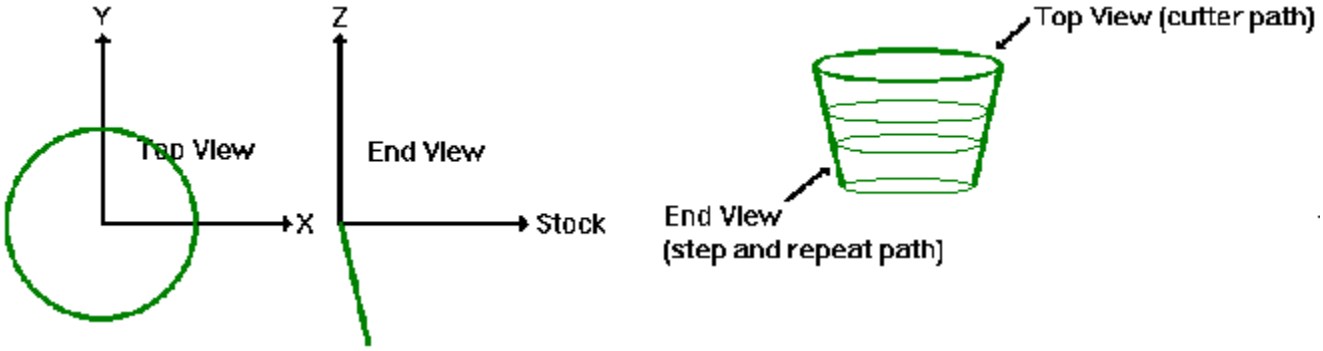
The Difference Between Top Views and End Views

When cutting a surface, you will probably want your cutter to follow a certain path. Once that path has been cut, the cutter generally steps over and down a bit and then follows a nearly identical path to the first. This process of stepping and repeating occurs over and over again, perhaps hundreds of times, until the desired surface emerges. The Top View describes to G-ZERO that primary XY cutter path. The End View describes to G-ZERO how much the tool steps over and how much the tool steps down between each pass. The tool steps down in Z. It steps over by adding (or deleting) extra STOCK each time.



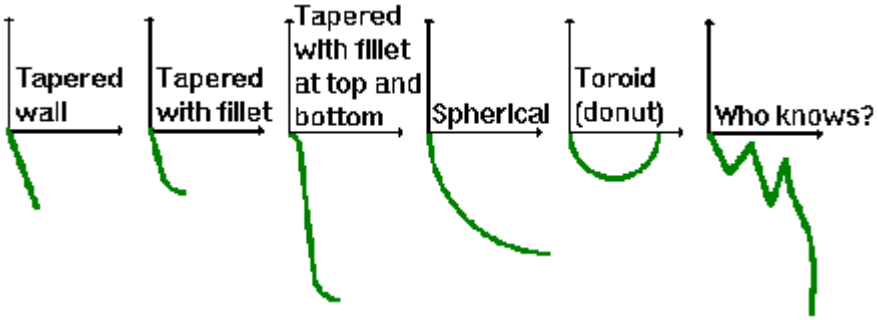
Since G-ZERO mostly thinks in XY, you just program your End Views like any other contour. +VIEW converts your XY contour into Z and STOCK.

To make the round tapered pocket example on page 3, our Top View is a ROUND and our End View is a line tapered down at 270+20 degrees:



The End View starts at 0,0 and travels mostly down and slightly to the right. This tells +VIEW that as each step moves down in Z, more stock must be left on the walls. End Views almost always move down in Z. And unless you have a special undercutting cutter, End Views usually grow in STOCK. In other words, End Views almost always stay in the X plus and Y minus.

Here are a few sample End Views:



Making +VIEW Do Something

We now have all the necessary know-how to rough and finish the 8" tapered pocket. Our rough tool will be a 1" sharp endmill:

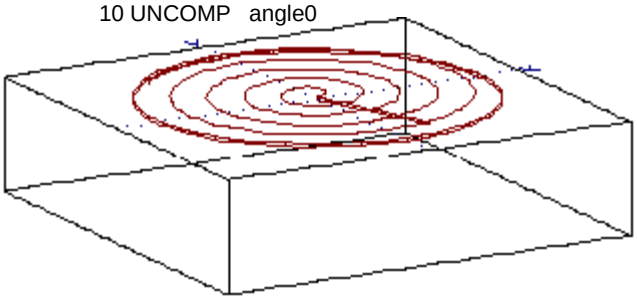
```
1 MAT'L   xmin-5 xmax5 ymin-5 ymax5 thk4 type1=ALUMALOY
2  TOOL 1 dia1 flutes2 type0 rad0
```

Next we define a Top View — a ROUND that will rough out the middle with a spiral:

```
3        Top View
4 ROUND   dia-8 x0 y0 thru0    <—— You must press the oh key to override and select a ROUND
```

Then we program the End View — a 290 degree line that goes 3 inches deep:

```
5        End View
6 COMP    angle0 cl/con1 lookahead0
7 POINT   x0 y0
8 LINE    angle290
9 POINT   x1.092 y-3   <—— There are ways to have G-ZERO calc this for you.
10 UNCOMP  angle0
```



The Top View and the End View sit together as 2 XY contours. +VIEW converts the End View into Z and STOCK info and then builds a surface.

Now, at long last, we select 38 +VIEW. G-ZERO then asks several questions:

WHAT PATTERN NUMBER CONTAINS THE VIEWS? 3.010

Remember from above that a Pattern is a separate file that contains geometry. In this case, however, our Views are not in a separate file, they are sitting right here on lines 3 thru 10. When such is the case (and it usually is for +VIEW), we simply glue the 3 together with the 10 by inputing 3.010. Note that G-ZERO will strip the last 0 and make it 3.01.

PUT RESULT INTO WHAT PATTERN NUMBER? 1

Since +VIEW may well produce thousands of lines of code, we need a place to store all this information. In this example, let's store the information in a place called "1".

STEPOVER HOW FAR PER PASS? .25

Since this is a roughing tool, let's take big steps. If we make the stepover too small, then our tape could get huge.

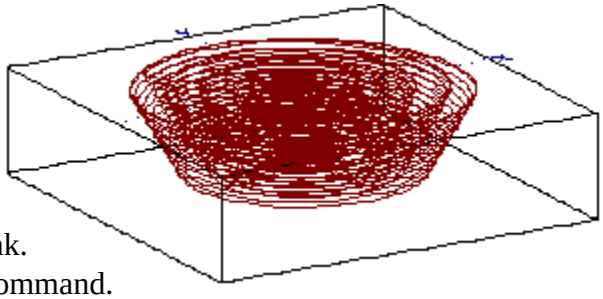
SHOULD THE TOOL RAMP BETWEEN EACH PASS? (1=yes,0=no) 1

For most simple +VIEWs, ramping can be turned on. Be careful though, if your Top View does not start and end in the same place, ramping could cause a crash!

IF MORE THAN 2 VIEWS, WOULD YOU LIKE BLENDING? (1=yes,0=no) 0

With just one Top View and one End View, your answer here is always 0.

```
11 +VIEW    in3.01 out1 step.25 ramp1 blend0
```

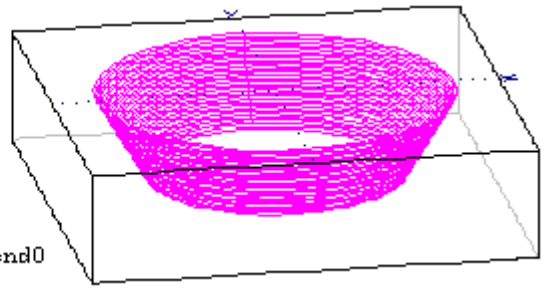


This little 11 line program is <u>almost</u> all it takes to generate a whole lot of roughing. If we were to post process this program right now, the results would be most unimpressive. In fact the program would be blank. The reason is simple — we never called out a MILL command. **The post ignores everything before the first MILL command** . Since the Views are for definition only, they will always occur before the first MILL. We use this area between a TOOL command and its first MILL command to hide things from the post processor. This area of source is called the "Dead Zone". So with a MILL added, the corrected source looks something like this:

```
11 MILL     zrapid.1 zcut0 passes1 zret.1 zf5 xyf10   <—— Zcut is safely above the cut.  +VIEW controls Z.
12 +VIEW    in3.01 out1 step.25 ramp1 blend0
```

Our finish tool will be a .25" ball endmill (don't forget the .125 rad). This time our Top View does not need to spiral from the center out, so we can use ROUND to generate one simple circle starting at the 90 degree position. Our End View is identical to the 20 degree taper we already input, so we can just REPEAT lines 6 thru 10. When calling up +VIEW be sure not to output on top of Pattern 1. Pattern 1 is already in use by the roughing tool.

```
13    TOOL 2 dia.25 flutes2 type0 rad.125
14           TV
15 ROUND     dia-8 x0 y0 thru90
16           EV
17 REPEAT    from6 thru10
18 MILL      zrapid.1 zcut0 passes1 zret.1
19 +VIEW     in14.017 out2 step.1 ramp1 blend0
```

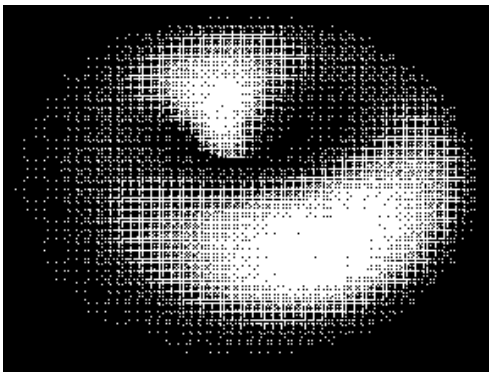


Congratulations! You have just created your first surface using +VIEW. Some people call this a "Surface of Revolution", others call it a "Cone" or "Conic" or "Simple Conic". Most of the time, we will refer to a shape like this as a "2 View". 2 Views can look surprisingly different from one another, but they are all made up of one Top View (which always comes first) plus one End View.

Let's summarize what we know about +VIEW

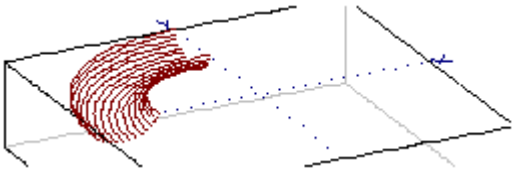
- Before using +VIEW, you must program 1 to 4 Views.
- A View is made up of several lines of source beginning with exactly one comment.
- A Top View describes the actual path of the cutter as it makes each pass.
- An End View gives +VIEW the Z and STOCK info it needs in order to step and repeat.
- Views are hidden in the dead zone before the first MILL command after a tool change.
- After the Views are strung together, call a MILL command to end the dead zone.
- The zcut question in the MILL command is not used. +VIEW controls the Z axis.
- +VIEW is choice 38.
- Tell +VIEW what source lines to process by giving the starting and ending line numbers in the form ###.### when +VIEW asks for the Pattern that contains the Views.
- +VIEW stores its surface in a Pattern.
- Patterns are separate files that are numbered from 1 to 9999.
- Patterns get overwritten by other progams.
- Patterns do not automatically update when Views get edited.
- To update Patterns, just redraw with the tool (F3).
- Give a reasonable stepover value. A small stepover like .0005 may cause too long a g-code file.
- Leave ramping off unless you are sure the path is clear.
- 2 Views don't use blending.

Surface of Revolution



2 View - Partial Torus

```
1 MAT'L    xmin-5 xmax5 ymin-5 ymax5 thk4
2   TOOL 1 dia.5 flutes2 type0 rad.25

3          tv
4 COMP     angle90 cl/con1 lookahead0
5 RADIUS   3 type1 x0 y0
6 UNCOMP   angle0

7          ev
8 COMP     angle270 cl/con1 lookahead0
9 RADIUS   -1 type1 x1 y0
10 UNCOMP  angle90

11 MILL    zrapid.1 zcut0 passes1 zret.1
12 +VIEW   in3.01 out1 step.2 ramp0 blend0
```

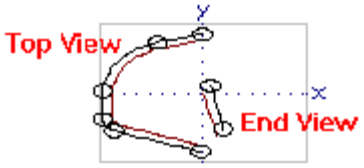


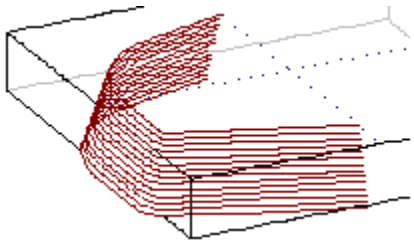
Note that we left ramping off in the +VIEW. Try turning ramping on and watch as your part gets sawed in half.

Surface of Translation
aka Swept Surface
aka Dragged Surface

2 View - Tapered walls on a contour

```
1 MAT'L    xmin-4 xmax4 ymin-4 ymax4 thk2.5
2   TOOL 1 dia.75 flutes2 type0 rad.375

3          tv - any old contour
4 COMP     angle180 cl/con1 lookahead0
5 POINT    x0 y-3
6 LINE     angle160
7 RADIUS   .25 type0 (xc-3.25 yc-1.551052)
8 LINE     angle90
9 RADIUS   2.5 type2 x-3.5 y2 (xc-1 yc.1142465)
10 POINT   x0 y3
11 UNCOMP  angle0

12         ev - simple taper
13 COMP     angle0 cl/con1 lookahead0
14 POINT   x0 y0
15 POINT   x.5 y-2.5
16 UNCOMP  angle0

17 MILL    zrapid.1 zcut0 passes1 zret.1
18 +VIEW   in3.016 out1 step.2 ramp0 blend0
```





Notice that even though our End View starts at X0Y0, the tool path starts quite a ways up in Z. By changing the COMP angle on line 13, you have complete control of the tool.

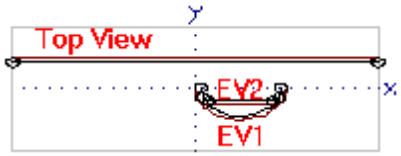
Ruled Surfaces

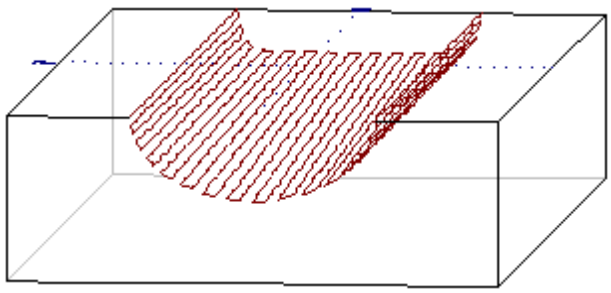
Up until now, all the shapes we have encountered share a common characteristic — they have a constant cross-section. If you slice a 2 View shape anywhere along the Top View, you will always see the same End View. But what if you want the End View to change?

+VIEW has the ability to "blend" from one End View to another. Just tack on an additional End View, and then tell +VIEW to turn blending on. The order of Views for a 3 View is always TV, EV1, EV2.

3 View — Straight Ruled Surface

```
1 MAT'L    xmin-4 xmax4 ymin-2 ymax2 thk2 type1
2   TOOL 1 dia.25 flutes2 type0 rad.125

3          tv is simple straight line
4 COMP     angle0 cl/con2 lookahead0
5 POINT    x-4 y1
6 POINT    x4 y1
7 UNCOMP   angle0

8          ev1 - the starting end view
9 COMP     angle270 cl/con1 lookahead0
10 RADIUS   -1 type1 x1 y0
11 UNCOMP   angle90

12         ev2 - at the part's far end
13 COMP     angle270 cl/con1 lookahead0
14 POINT    x0 y0
15 LINE     angle280
16 RADIUS   -.25 type0 (xc.2979384 yc-.25)
17 LINE     angle0
18 POINT    x1 y-.5
19 LINE     angle0
20 RADIUS   -.25 type0 (xc1.702062 yc-.25)
21 LINE     angle80
22 POINT    x2 y0
23 UNCOMP   angle90

24 MILL     zrapid.1 zcut0 passes1 zret.1 zf5
25 +VIEW    in3.023 out111 step.1 ramp1 blend1
```





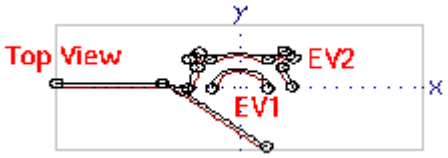
Ramp has extra power when used with a 3 View. It causes +VIEW to zigzag back and forth instead of lifting the tool and rapiding back to the start point.

3 View - Bent Ruled Surface

1 MAT'L xmin-4 xmax4 ymin-2 ymax2 thk4
2 TOOL 1 dia.25 flutes2 type0 rad.125

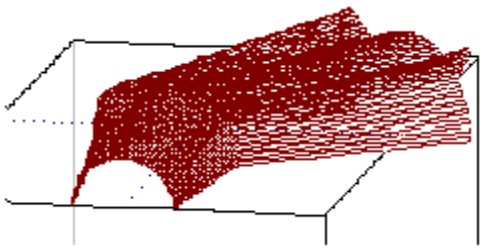
3 tv - drive along a bent line
4 COMP angle0 cl/con1 lookahead0
5 POINT x-4 y0
6 LINE angle0
7 RADIUS .5 type0 (xc-1.707106 yc-.5)
8 LINE angle315
9 POINT x.5 y-2
10 UNCOMP angle315

11 ev1 - one the rare times we don't start at X0Y0
12 COMP angle90 cl/con1 lookahead0
13 RADIUS .5 type1 x0 y0
14 UNCOMP angle270



15 ev2
16 COMP angle90 cl/con1 lookahead0
17 RADIUS 1 type1 x0 y0
18 RADIUS -.125 type0 (xc-.9219728 yc.6446636)
19 ROTATE angle45 xpiv0 ypiv0
20 LINE angle80
21 RADIUS .02 type2 x0 y1.5 (xc-.9792187 yc.9792187)
22 LINE angle280
23 ROTATE angle-45 xpiv0 ypiv0
24 RADIUS -.125 type0 (xc-.6446635 yc.9219729)
25 RADIUS 1 type1 x0 y0
26 RADIUS -.125 type0 (xc.6270399 yc.9340482)
27 LINE angle80
28 RADIUS .02 type1 x0 y1.5
29 LINE angle280
30 RADIUS -.125 type0 (xc.9340482 yc.6270397)
31 ROTATE angle0 xpiv0 ypiv0
32 RADIUS 1 type1 x0 y0
33 UNCOMP angle270

34 MILL zrapid.1 zcut0 passes1 zret.1
35 +VIEW in3.033 out12 step.05 ramp1 blend1

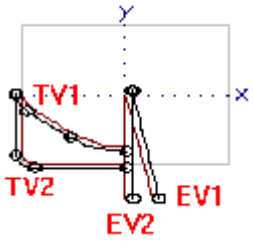


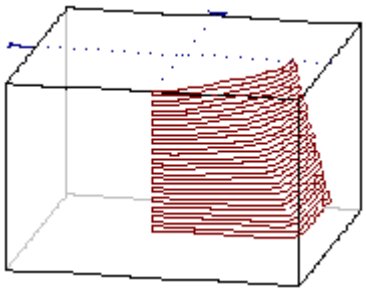
Blended Surface

Some blended shapes require more control than 3 Views can offer. The edges of a 3 View are a linear interpolation from one End View to the next. When you need to control the growing and shrinking of your surface along the edges, you must give an extra Top View. The order of Views for a 4 View is always TV1, EV1, EV2, TV2.

4 View — Side Patch

```
1 MAT'L    xmin-2 xmax2 ymin-2 ymax2 thk4
2   TOOL 1 dia.25 flutes2 type0 rad.125

3         tv1 - rounded at the top
4 COMP     angle180 cl/con1 lookahead0
5 RADIUS   1.5 type1 x0 y0
6 RADIUS   .5 type1 x-1.5 y0
7 UNCOMP   angle90

8         ev1
9 COMP     angle0 cl/con1 lookahead0
10 POINT    x0 y0
11 POINT    x.5 y-3
12 UNCOMP   angle270

13         ev2 - steeper than 1st ev
14 COMP     angle0 cl/con1 lookahead0
15 POINT    x0 y0
16 POINT    x0 y-3
17 UNCOMP   angle270

18         tv2 - squared at the bottom
19 COMP     angle180 cl/con1 lookahead0
20 POINT    x0 y-2
21 RADIUS   .25 type2 x-2 y-2 (xc-1.75 yc-1.75)
22 POINT    x-2 y0
23 UNCOMP   angle90

24 MILL     zrapid.1 zcut0 passes1 zret.1 zf5 x
25 +VIEW    in3.023 out1 step.1 ramp1 blend1
```





+TILT

Choice 39 in G-ZERO's Mill System menu is called +TILT (cross tilt). This command is an optional addition to G-ZERO. To call it up, you may enter 39 directly, or you may click the item on the menu.

Patterns or Lines Are Input

Before you can use +TILT to actually create a surface, you must first give G-ZERO some information to work from. Most of the time, you will wish to tilt a surface that you just created in +VIEW. Since +VIEW saves its output to a separate file called a Pattern, +TILT works best when you give it a Pattern number as input. Sometimes, however, you may want to tilt something simpler. This could be geometry like COMP, POINT, RADIUS, LINE, UNCOMP or occasionally RECT or ROUND or maybe even ROUGH. To tilt a non-pattern (say lines 10 through 20 of your source), just enter ###.### (10.020) instead of a Pattern number. Note that G-ZERO shortens 10.020 to 10.02 on the display screen.

Patterns Are Output

Once you answer +TILT's many questions, the tilting takes place immediately on-screen. At the same time, all the hundreds (or thousands or perhaps even millions) of surface points are saved in a file on your disk. This file is called a "Pattern". Patterns are similar to Widgets (macros), but they are compressed and therefore use less disk space and execute much faster.

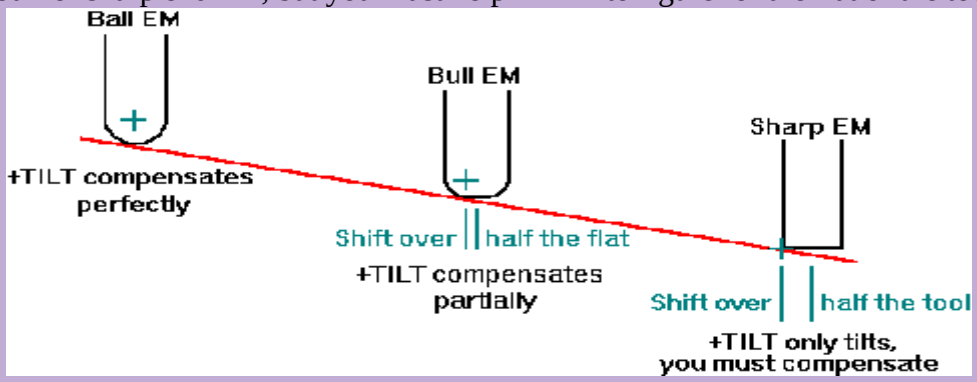
- Patterns are numbered, starting at 1 and continuing all the way to 9999.
- Patterns are very easily overwritten by other programs.
- Unlike other objects in G-ZERO, patterns do not automatically update when a tool diameter or some geometry changes.

In other words, Patterns are powerful and they are fragile. Luckily, it is very easy to regenerate a pattern — just redraw your source with the tool. If you forget this simple rule, your final G-code might not reflect your last edits. Worse yet, your final G-code could contain data from another completely different part.

So always be sure to press the F3 key before posting an un-new 3D source program.

Think Ball Endmill

+TILT works best when you use it with ball endmills. G-ZERO *does* allow you to tilt some geometry using a bull or sharp endmill, but you must help +TILT to figure for the flat of the tool.



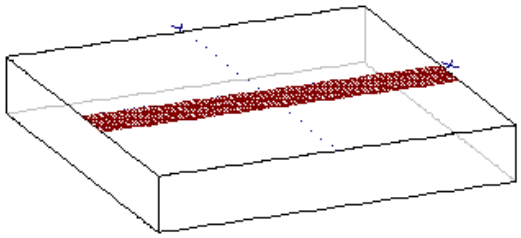
Making +TILT Do Something

Let's start by cutting a large bevel on the corner of a part. Our tool will be a .5" ball endmill:

```
1 MAT'L    xmin-4 xmax4 ymin-4 ymax4 thk2 type1=ALUMALOY
2   TOOL 1 dia.5 flutes2 type1 rad.25 *** ball em
```

Next we define a rectangular rough area with a stepover that is much smaller than normal:

```
3 ROUGH    stk0 stp.05 angle0 cleanup0 <—— You must press the oh key to override and select a ROUGH
4 POINT    x4 y0 f5                    <—— You must press the oh key to override and select a POINT
5 POINT    x-4 y0 f5
6 POINT    x-4 y-1 f5
7 POINT    x4 y-1 f5
8 ROUGH    stk0 stp.05 angle0 cleanup0
```



Now we select 39 +TILT. G-ZERO then asks several questions:

WHAT LINE NUMBERS (or PATTERN) WOULD YOU LIKE TO TILT? 3.008
Remember from above that a Pattern is a separate file that contains geometry. In this case, however, our geometry is not in a separate file, it is sitting right here on lines 3 thru 8. When such is the case, we simply glue the 3 together with the 8 by inputting 3.008.

PUT RESULT INTO WHAT PATTERN NUMBER? 1
Since +TILT may well produce thousands of lines of code, we need a place to store all this information. In this example, let's store the information in a place called "1".

TILT HOW MANY DEGREES TOWARD YOU? (minus for away) 30
To answer this question, think about the stick a pilot uses to steer an airplane. Pulling the stick 30 degrees toward you would lift the nose of the plane 30 degrees.

TILT HOW MANY DEGREES TO THE LEFT? (minus to the right) 0
We want no left/right tilt, so enter 0. If you do need to roll, then remember: all tilting toward/away is calculated *before* left/right tilting.

ROTATE HOW MANY DEGREES IN THE STANDARD XY PLANE? 0
This is just like the rotation caused by choice 16) ROTATE. In this case we say 0.

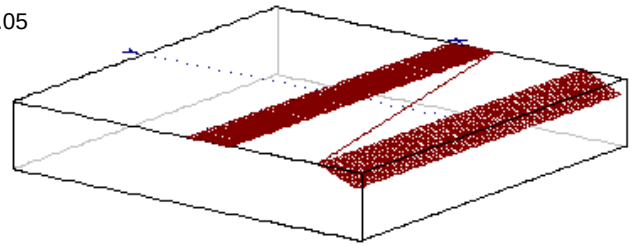
HOW FAR IN X WOULD YOU LIKE THE PATTERN TRANSLATED? 0
Since we want no shift in X, we enter 0.

HOW FAR IN Y? -3
To shift down from the center of the part to the near edge, we enter -3 inches.

HOW FAR IN THE Z AXIS? 0
No Z shift.

BREAK ARCS INTO WHAT SIZE STEPS? .05
Our shape is made up exclusively of points, so this answer really does not matter. However, some value MUST be given, so enter .05 here.

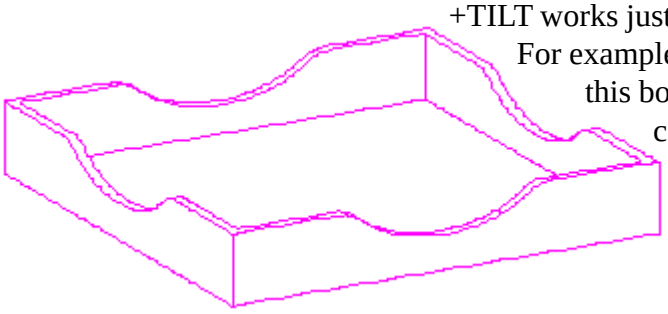
```
9 +TILT    in3.008out1pitch30roll0rot0x0y-3z0stp.05
```



This little 9 line program is *almost* all it takes to generate a beveled cut. Notice the line dragging between the definition geometry and the tilted geometry. This line should be a clue. If we were to post process the program right now, the results would be most unimpressive. In fact the program would be blank. The reason is simple — we never called out a MILL command. **The post ignores everything before the first MILL command** . Since the geometry is for definition only, it will usually occur before the first MILL. We use this area between a TOOL command and its first MILL command to hide things from the post processor. This area of source is called the "Dead Zone". So with a MILL added, the corrected source looks like this:

```
9 MILL     zrapid.1 zcut0 passes1 zret.1 zf5 xyf10   <—— Zcut is safely above the cut.  +TILT controls Z.
10 +TILT    in3.008out1pitch30roll0rot0x0y-3z0stp.05
```

Using +TILT On Arcs



+TILT works just as well on geometry that is made up of radiuses. For example, to cut the XZ and YZ contours in the walls of this box, we simply program in XY and then let +TILT convert. To cut a shape like this, a very large ball endmill makes the going easier. Smaller endmills might require several passes in order to make the part's top flat.

```
1 MAT'L    xmin-4 xmax4 ymin-4 ymax4 thk2
2   TOOL 1 dia1 flutes2 type1 rad.5 *** big ball em
```

Next, we define an XY contour (3 radii):

```
3 COMP     angle0 cl/con1 lookahead0
4 RADIUS   .5 type1 x-2 y-.5
5 RADIUS   -2 type0 (xc0 yc1)
6 RADIUS   .5 type1 x2 y-.5
7 UNCOMP   angle0
```

Then we end the dead zone with a MILL command:

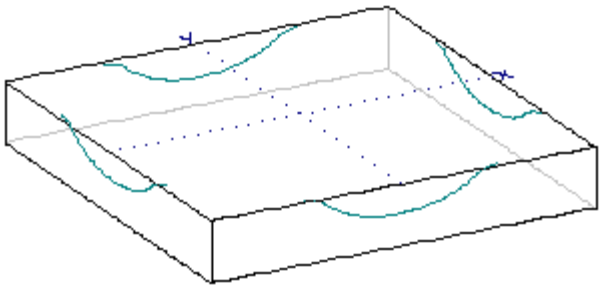
```
8 MILL     zrapid.1 zcut0 passes1 zret.1 zf5 xyf5
```

The near edge is cut by pitching +90 degrees and shifting -3.9 inches in Y. +TILT breaks the arcs into points. The .05 stepover breaks these arcs into almost a hundred points. Be careful when choosing a stepover — extremely small stepovers may create huge G-code files:

```
9 +TILT    in3.007out1pitch90roll0rot0x0y-3.9z0stp.05
```

Another MILL rapids our tool to the next cut. Be sure not to overwrite Pattern 1:

```
10 MILL    zrapid.1 zcut0 passes1 zret.1 zf5 xyf5
11 +TILT   in3.007out2pitch90roll0rot0x0y3.9z0stp.05      <—— far edge
12 MILL    zrapid.1 zcut0 passes1 zret.1 zf5 xyf5
13 +TILT   in3.007out3pitch90roll0rot90x-3.9y0z0stp.05  <— add rotation+pitch for left edge
14 MILL    zrapid.1 zcut0 passes1 zret.1 zf5 xyf5
15 +TILT   in3.007out4pitch90roll0rot90x3.9y0z0stp.05     <— right
```



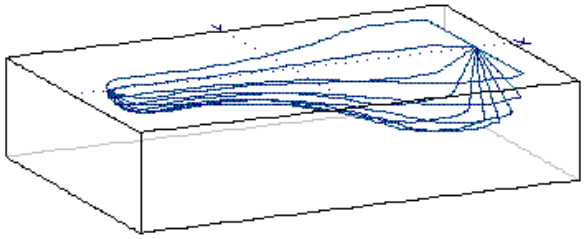
Let's summarize what we know about +TILT

- Before using +TILT, you must create some geometry to tilt.
- The geometry is made up of several lines of source or another 3D command that creates a Pattern.
- Geometry is hidden in the dead zone before the first MILL command after a tool change.
- After the geometry, call a MILL command to end the dead zone.
- The zcut question in the MILL command is safely above the part. +TILT controls the Z axis after that.
- +TILT is choice 39.
- Enter the source lines to process by giving the starting and ending line numbers in the form ###.### when +TILT asks for the Line Numbers to tilt.
- +TILT stores its surface in a Pattern.
- Patterns are separate files that are numbered from 1 to 9999.
- Patterns get overwritten by other programs.
- Patterns do not automatically update when Views get edited.
- To update Patterns, just redraw with the tool (F3).
- Tilt the surface toward you (+) or away from you (-) with the Pitch angle.
- Tilt the surface left (+) or right (-) with the Roll angle. Roll is calculated after Pitch.
- Rotate the surface after it has been pitched and rolled with the Rot angle.
- Give a reasonable stepover value. A small stepover like .0005 may cause too long a G-code file.

+TILT - Surface of Revolution

```
1 MAT'L    xmin-3 xmax3 ymin-2 ymax2 thk2
2   TOOL 1 dia.25 flutes2 type1 rad.125 *** ball em
3 MILL     zrapid.1 zcut0 passes1 zret.1
4 COMP     angle270 cl/con1 lookahead0
5 RADIUS   -.5 type1 x-2 y0
6 RADIUS   5 type0 (xc-1.5 yc-5.477226)
7 RADIUS   -1.5 type1 x2 y0
8 RADIUS   -.0001 type0 (xc2.4999 yc-1.414143)
9 LINE     angle90
10 POINT    x2.5 y0
11 UNCOMP   angle90

12 +TILT    in3.011out1pitch20roll0rot0x0y0z0stp.05
13 +TILT    in3.011out2pitch40roll0rot0x0y0z0stp.05
14 +TILT    in3.011out3pitch60roll0rot0x0y0z0stp.05
15 +TILT    in3.011out4pitch80roll0rot0x0y0z0stp.05
16 +TILT    in3.011out5pitch100roll0rot0x0y0z0stp.05
17 +TILT    in3.011out6pitch120roll0rot0x0y0z0stp.05
18 +TILT    in3.011out7pitch140roll0rot0x0y0z0stp.05
19 +TILT    in3.011out8pitch160roll0rot0x0y0z0stp.05
20 +TILT    in3.011out9pitch180roll0rot0x0y0z0stp.05
```

<— Lines 3 thru 11 *include* MILL command



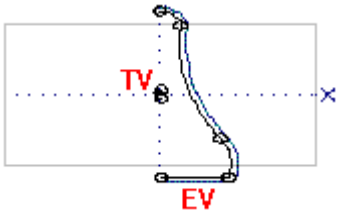
+TILT with +VIEW - A Powerful Combination

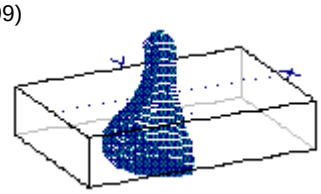
+TILT by itself can produce a fair number of different shapes. The problem, as we saw in the previous example, is that you may have to call +TILT over and over again to run the tool across an entire surface. +VIEW is especially good at stepping and repeating the tool path, so when we use +TILT on a +VIEW Pattern, almost any shape is attainable.

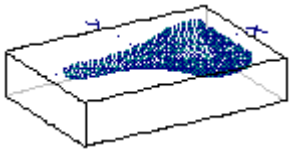
+TILT with 2 View - Surface of Revolution

In this example, we use +VIEW to create the geometry. The arcs are stored in Pattern 1. Then we tell +TILT to input Pattern 1 and output Pattern 2:

```
1 MAT'L    xmin-3 xmax3 ymin-2 ymax2 thk2 type1=ALUMALOY
2   TOOL 1 dia.25 flutes2 type1=CARBIDE MILL rad.125 *** ball em
3          Top View
4 COMP     angle180 cl/con1 lookahead0
5 RADIUS   .0001 type1 x0 y0
6 UNCOMP   angle0
7          End View (the profile is stood on end)
8 COMP     angle0 cl/con1 lookahead0
9 POINT    x0 y-2.5
10 LINE    angle0
11 RADIUS   -.0001 type0 (xc1.414143 yc-2.4999)
12 RADIUS   -1.5 type1 x0 y-2
13 RADIUS   5 type0 (xc5.477226 yc1.5)
14 RADIUS   -.5 type1 x0 y2
15 UNCOMP   angle180

16 +VIEW    in3.015 out1 step.1 ramp0 blend0

17 MILL     zrapid.1 zcut0 passes1 zret.1 zf5 xyf5
18 +TILT    in1out2pitch0roll90rot0x0y0z0stp.05
```

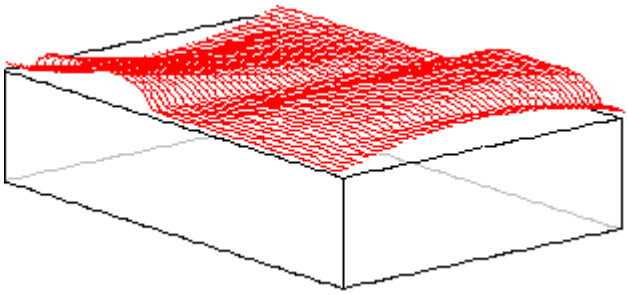






+TILT with 4 View — Surface Patch

```
1 MAT'L    xmin-2 xmax2 ymin-3 ymax3 thk2
2   TOOL 1 dia.25 flutes2 type1 rad.125 *** ball em

3          Top View 1
4 COMP     angle270 cl/con1 lookahead0
5 POINT    x0 y3
6 LINE     angle270
7 RADIUS   -.25 type0 (xc.25 yc2.653835)
8 RADIUS   .45 type1 x0 y2
9 RADIUS   .45 type1 x0 y1
10 RADIUS   -.25 type0 (xc.25 yc.3461652)
11 LINE     angle270
12 POINT    x0 y-3
13 UNCOMP   angle270

14         End View 1
15 COMP     angle99.99 cl/con1 lookahead0
16 POINT    x0 y0
17 RADIUS   -5 type0 (xc4.582576 yc-2)
18 POINT    x0 y-4
19 UNCOMP   angle99.99

20         End View 2
21 COMP     angle99.99 cl/con1 lookahead0
22 POINT    x0 y0
23 RADIUS   5 type0 (xc-4.582576 yc-2)
24 POINT    x0 y-4
25 UNCOMP   angle99.99

26         Top View 2
27 COMP     angle270 cl/con1 lookahead0
28 POINT    x0 y3
29 LINE     angle270
30 RADIUS   -.25 type0 (xc.25 yc-.3461652)
31 RADIUS   .45 type1 x0 y-1
32 RADIUS   .45 type1 x0 y-2
33 RADIUS   -.25 type0 (xc.25 yc-2.653835)
34 LINE     angle270
35 POINT    x0 y-3
36 UNCOMP   angle270

37 +VIEW    in3.036 out1 step.1 ramp0 blend.1

38 MILL     zrapid.1 zcut0 passes1 zret.1
39 +TILT    in1out2pitch0roll90rot0x-2y0z0stp0
```



+TRIM

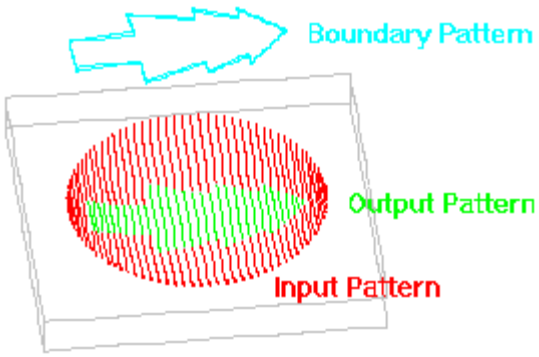
Choice 40 in G-ZERO's Mill System menu is called +TRIM (cross trim). This command is an optional addition to G-ZERO. To call it up, you may enter 40 directly, or you may click the item from the menu. This command is used to trim away all unwanted data from a Surface. The following discussion assumes the reader is familiar with +VIEW, +TILT and Patterns.

Patterns of *Points* Are Input

+TRIM is a finicky eater. You must store the surface or geometry to trim in a Pattern before calling +TRIM. Furthermore, +TRIM hates arcs, so you must convert any arcs in a Pattern to points. Use the +TILT command to break the arcs into points. The stepover question in +TILT allows you to determine point density.

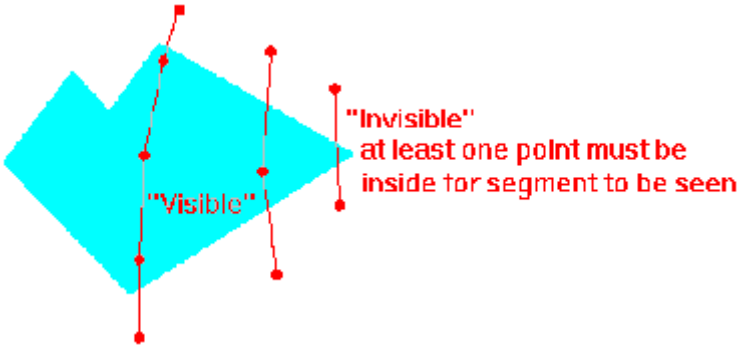
Plus An Additional Boundary Pattern

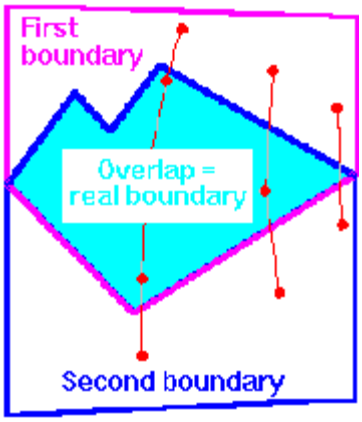
+TRIM takes one Pattern (a surface) and then uses another Pattern (a boundary) as a sort of cookie cutter on the first Pattern:



It Can Only Trim What It Can See

For +TRIM to "see" a line segment, the segment must pass from inside the boundary to outside (or vice versa):





To help +TRIM see things, just extend your boundary into two separate *overlapping* boundaries and then use +TRIM twice:

Making +TRIM Do Something

Let's take a chomp out of a donut. First we need to create a donut shaped (torus) surface with a .25" ball endmill:

```
1 MAT'L    xmin-2 xmax2 ymin-2 ymax2 thk2
2   TOOL 1 dia.25 flutes2 type1 rad.125 *** ball em
3          Top View
4 COMP     angle90 cl/con1 lookahead0
5 RADIUS   1 type1 x0 y0
6 UNCOMP   angle90
7          End View
8 COMP     angle90 cl/con1 lookahead0
9 RADIUS   .5 type1 x0 y0
10 UNCOMP   angle270
11 +VIEW    in3.01 out1 step.1 ramp0 blend0

12         Convert Surface from arcs to points
13 +TILT    in1out2pitch0roll0rot0x0y0z0stp.05          <— No rotation, just breaking surface.

14         Now the boundary (bite shaped)
15 COMP     angle45 cl/con1 lookahead0
16 RADIUS   .625 type1 x-1.5 y0
17 UNCOMP   angle135

18         Convert boundary from an arc to points
19 +TILT    in15.017out3pitch0roll0rot0x0y0z0stp.05   <— No rotation, just breaking boundary.
20 MILL     zrapid.1 zcut0 passes1 zret.1
```

Now, we are finally ready to call 40) +TRIM:

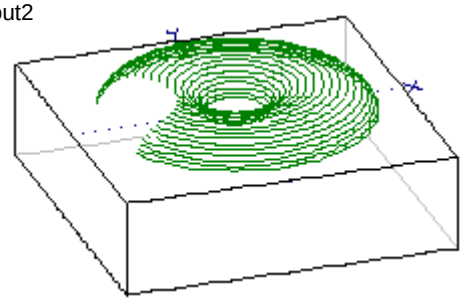
WHAT PATTERN NUMBER WOULD YOU LIKE TRIMMED? 2
Pattern 2 contains the surface points.

PUT RESULT INTO WHAT PATTERN NUMBER? 4
Next available Pattern is 4.

WHAT PATTERN NUMBER CONTAINS THE BOUNDARY? 3
Pattern 3 contains boundary points

WOULD YOU LIKE TO STAY: 1) inside OR 2) outside THE BOUNDARY? 2
To keep data that is inside the bite, enter 1. In our case we want all the points that are outside the bite.

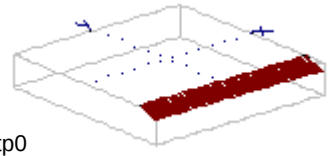
```
21 +TRIM    in2 out4 bound3 in/out2
```



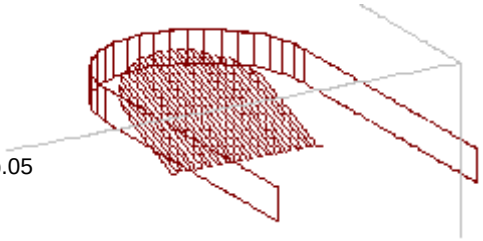
+TRIM an angled plane

Using the first example in the +TILT manual, let's trim a circular section:

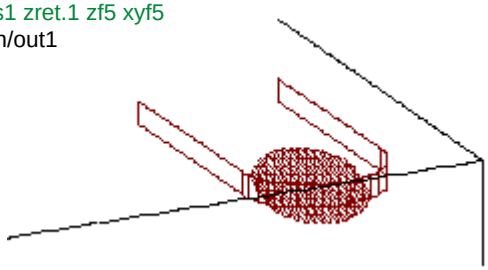
```
1 MAT'L    xmin-4 xmax4 ymin-4 ymax4 thk2 type1=ALUMALOY
2   TOOL 1 dia.25 flutes2 type1=CARBIDE MILL rad.125 *** ball em
3 ROUGH    stk0 stp.025 angle0 cleanup0
4 POINT    x4 y0 f5
5 POINT    x-4 y0 f5
6 POINT    x-4 y-1 f5
7 POINT    x4 y-1 f5
8 ROUGH    stk0 stp.025 angle0 cleanup0
9 +TILT    in3.008out1pitch30roll0rot0x0y-3z0stp0
```



```
10          Now the first semi-circular boundary
11 COMP     angle90 cl/con1 lookahead0
12 POINT    x3.5 y-5
13 RADIUS   -.5 type1 x3 y-3.5
14 POINT    x2.5 y-5
15 UNCOMP   angle270
16 +TILT    in10.015out2pitch0roll0rot0x0y0z0stp.05
17 +TRIM    in1 out11 bound2 in/out1
```



```
18          Now the second boundary
19 COMP     angle270 cl/con1 lookahead0
20 POINT    x2.5 y-2
21 RADIUS   -.5 type1 x3 y-3.5
22 POINT    x3.5 y-2
23 UNCOMP   angle90
24 +TILT    in18.023out3pitch0roll0rot0x0y0z0stp.05
25 MILL     zrapid.1 zcut0 passes1 zret.1 zf5 xyf5
26 +TRIM    in11 out21 bound3 in/out1
```



Let's summarize what we know about +TRIM

- Before using +TRIM, you must create some geometry to trim.
- The geometry is always stored in a Pattern.
- The Pattern can only contain points. Use +TILT to convert a Pattern containing arcs into one with only points.
- Before using +TRIM, you must also create a boundary Pattern that contains only points.
- If line segments in the surface pattern are too long, then +TRIM may not see them crossing the boundary . In that case, you must set-up 2 +TRIMs with overlapping boundaries.
- +TRIM is choice 40.

+PROJECT

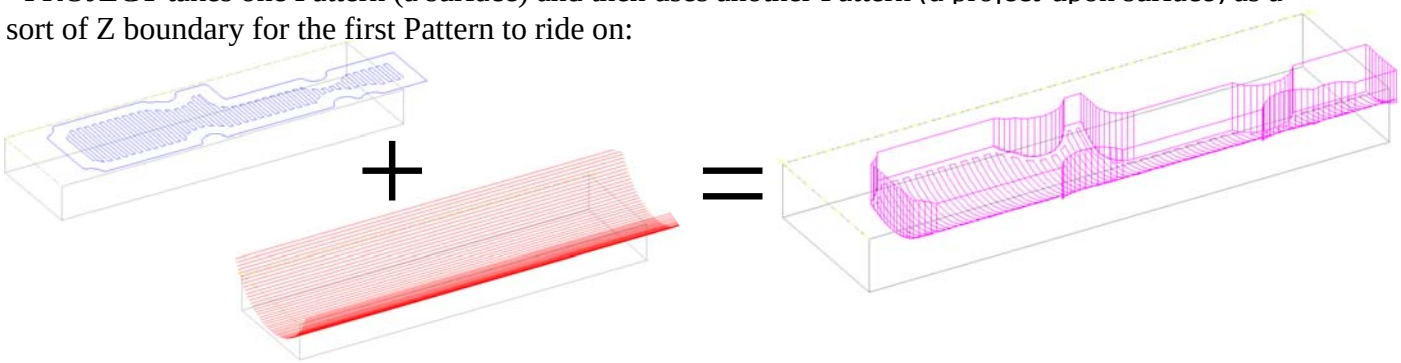
Choice 41 in G-ZERO's Mill System menu is called +Project. This command is an optional addition to G-ZERO. To call it up, you may enter 41 directly, or you may click the item from the menu. This command is used to modify the Z information in a Pattern, so that it "rides on" another Pattern. The following discussion assumes the reader is familiar with +VIEW and Patterns (see earliest pages in this manual).

Patterns or Lines Are Input

Before you can use +PROJECT to create a surface, you must first give G-ZERO some information to work from. Often, you will wish to project a surface that you just created in +VIEW. Since +VIEW saves its output to a separate file called a Pattern, +PROJECT works best when you give it a Pattern number as input. Sometimes, however, you may want to project something simpler. This could be geometry like COMP, POINT, RADIUS, LINE, UNCOMP or occasionally RECT or ROUND or maybe even ROUGH. To project a non-pattern (say lines 10 through 20 of your source), just enter ###.### (10.020) instead of a Pattern number. Note that G-ZERO shortens 10.020 to 10.02 on the display screen.

Plus An Additional "Project Upon" (boundary) Pattern

+PROJECT takes one Pattern (a surface) and then uses another Pattern (a project-upon surface) as a sort of Z boundary for the first Pattern to ride on:



```
1  MAT'L    xmin0 xmax2.7 ymin-.778 ymax0 thk.275...
2    TOOL 1 dia.1875 flutes2 type1 rad.0935 ...
3
4          XY POCKET DATA TO PROJECT
5  ROUGH    stk.02 stp.03 angle180 cleanup-1
6    COMP     ang90 cl/cn1 lookahead1 lr0 la0
7    POINT    x2.9 y-.5
8    RADIUS   -.031 type1 x2.387 y-.32
  ...
33   POINT    x2.9 y-.5
34   UNCOMP   angle90 lr0 la0
35  ROUGH    stk.02 stp.03 angle180 cleanup-1

36         tv - PROJECT UPON
37  COMP     ang0 cl/cn1 lookahead0 lr0 la0
38  POINT    x0 y0
39  POINT    x4 y0
40  UNCOMP   angle0 lr0 la0
41           ev - PROJECT UPON
42  COMP     ang-70 cl/cn1 lookahead0 lr0 la0
43  RADIUS   -.5 type1 x-.389 y.255
44  UNCOMP   angle70 lr0 la0
45   +VIEW    in36.044 out1177 step.025 ramp0 blend0
46  MILL     zrapid.1 zcut.01 passes1 zret.1 zf50 xyf30
47   +PROJ    in4.035 out1178  surf1177 uponly0
```

Here are the questions asked by +PROJECT:

WHAT PATTERN NUMBER WOULD YOU LIKE PROJECTED? 4.035

In our example, lines 4 thru 35 contain the XY shape to be modified (projected).

SPECIAL CASES:

- You may wish to bend or warp one 3D surface with another, for example to cut a multi-step bezel around a large spherical radius. Or a filet radius around the bottom of a 3D pocket. If so, then create a 3D surface using +View or +Tilt (or both), and enter its pattern number here.
- In rare cases, you may wish to collide one surface into another, by pushing one surface up through another, lifting just a corner, for example. If so, then create a 3D surface using +View or +Tilt (or both), and enter its pattern number here.

PUT RESULT INTO WHAT PATTERN NUMBER? 1178

In this example, every Pattern except 1177 is available.

SPECIAL CASE: The input pattern (previous question) may contain arcs. If so, G-ZERO will break the arcs every .05 inches so +Project can process and output XYZ points. To change .05 to another stepover, enter your step value here by adding it to the pattern number. For example to output pattern 3 with an arc stepover of .025, enter 3.025.

WHAT PATTERN NUMBER CONTAINS THE SURFACE? 1177

Pattern 1177 contains +VIEW data. A simple "2 View" descrbing a cylinder laid on its side.

SPECIAL CASES:

- +Project must break everything into points in order to do the math. Arcs in the project-upon-surface are automatically broken at 1 degree stepovers. To increase precision (especially useful for surfaces made up of very large arcs), enter angles smaller than 1 by adding the decimal angle to this pattern number. For example, if your surface is stored in pattern 7, and you want arcs broken every .25 degrees, enter 7.25 here. Note that smaller stepovers greatly increase the time +Project takes to do its calculations.
- If +Project lets some points 'fall through the surface', then it means the routine is confused. Force it to use a different surface recognition algorithm by entering this pattern as a minus number. For example to output pattern 7, enter -7.

HOW SHOULD Z BE HANDLED? (1=up only, 0=full project) 0

Most common answer is 0.

SPECIAL CASES:

- In some cases, you may wish to bend or warp one 3D surface with another, for example to cut a multi-step bezel around a large spherical radius. Or a filet radius around the bottom of a 3D pocket. If so, then enter a small number like .0001 or -.0001 here. If the tool seems to be riding too high or too low in Z, adjust this number down or up. Be sure not to adjust to exactly 0 or 1.
- Warping can also work very well when engraving on 3D surfaces. Just enter the depth you'd like your engraving tool to cut. For example, -.01 is a common engraving depth.

Let’s summarize what we know about +PROJECT

- Before using +PROJECT, you must create some XY geometry to project.
- Before using +PROJECT, you must also create a 'project upon' surface using +VIEW.
- +PROJECT is choice 41.
- +PROJECT's last question makes it possible to lift just a section of your input shape (enter 1), or to 'engrave' an input shape (enter small negative value), or to 'warp' an input surface (enter any number other than 0 or 1) around the project upon surface, so that filleting a complex surface, for example, becomes relatively simple.

For more info, check out the following examples in C:\Mill\Examples\3DBB:

Pool.M
Pool+Letter.M
Bezel.M
Yikes.M
Saddleback.M
Mouse.M
Donut.M
Fly Wheel.M