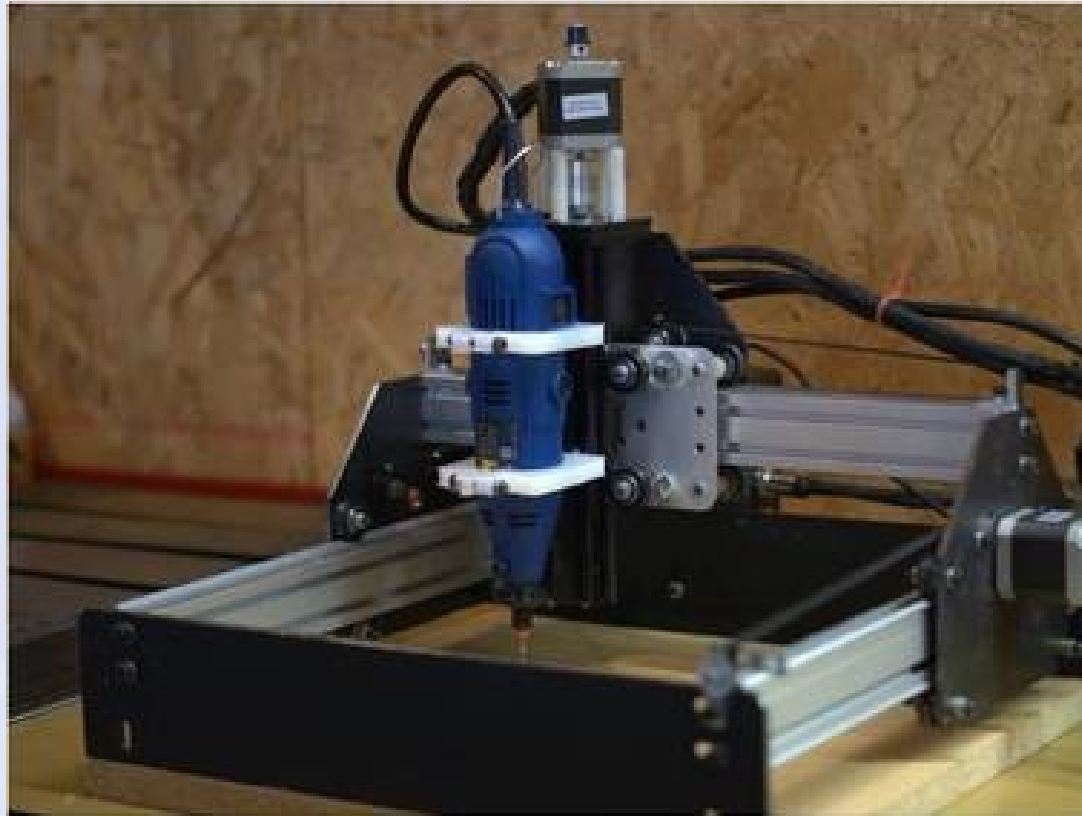


Shapeoko CNC router



Brian Paavo, 9 Aug 2012, Dspace.org.nz

Thank you Paul!

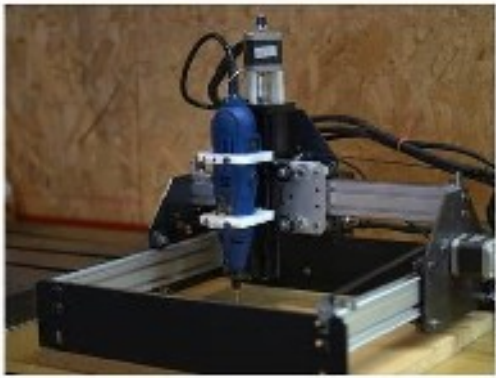
INVENTABLES Shopping Cart (0) Account Log Out

[HARDWARE](#) [3D PRINTING MATERIALS](#) [LASER CUTTING](#) [RAW MATERIALS](#) [GIFT CERTIFICATES](#)

[Home](#) [Partners](#) [Tools](#) [MACHINES](#) [CNC MILL KITS](#) [Shapeoko](#)

CNC Mill Kits - Shapeoko

3 open source kits to build a machine for milling plastics, woods, and metals



PLACE YOUR PRE-ORDER

19	50		
Price	Per kit delivery		
Delivery scheduled to start in			
21	5	18	42
Days	Hours	Days	Hours

The fine print: This product is an ongoing project and is not a finished item. We cannot guarantee the exact amount of pre-orders and only while the supply is limited. We will have some pre-orders that we will not include in the included.

PICK A TYPE

Price	SKU	Type
\$259.00	2014JUL	Mechanical Kit (requires you provide electronics to run the CNC)
\$269.00	2014JUL	Full Kit (everything you need to run your machine)
\$289.00	2014JUL	High Precision Kit (full kit - stable and the most accurate)

Qty:

[View all CNC Mill Kits](#)

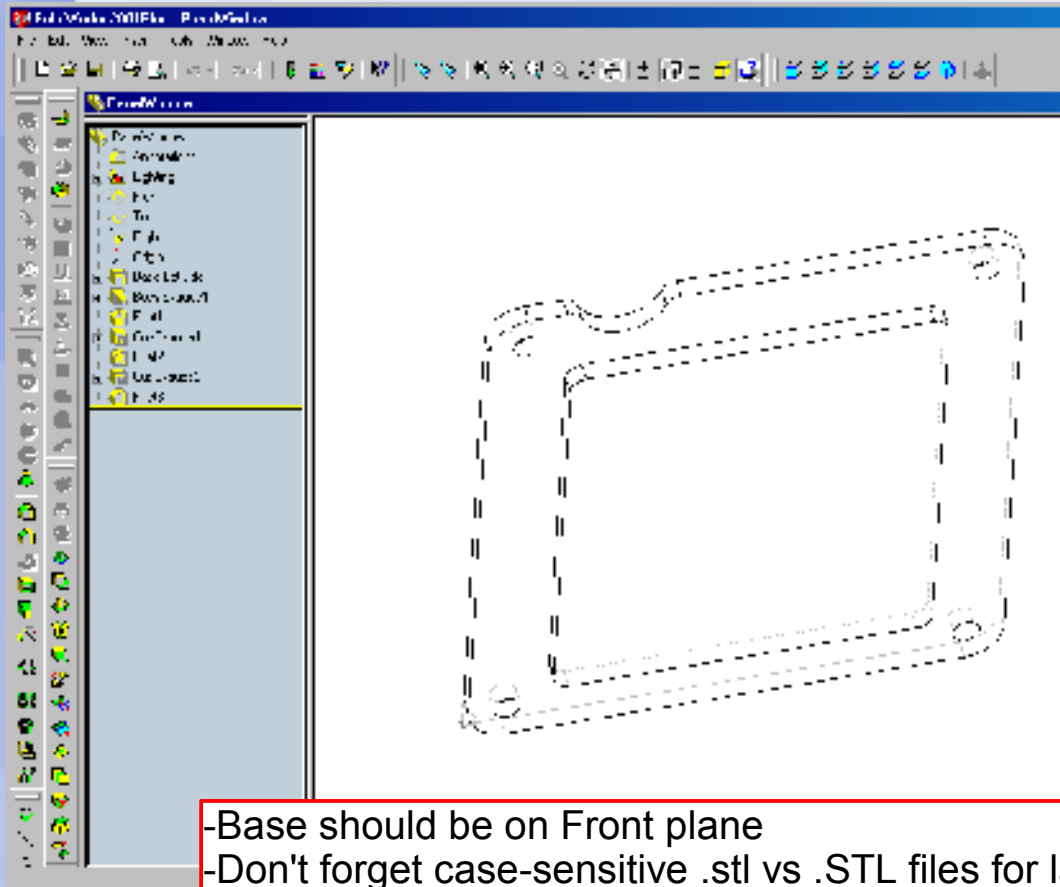
© 2014 Inventables LLC. All rights reserved. [Privacy Policy](#) [Terms of Service](#)

The Chat

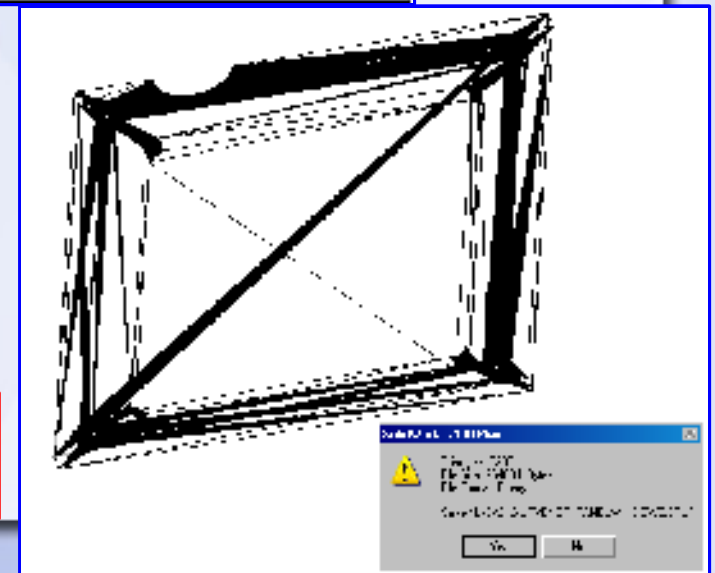
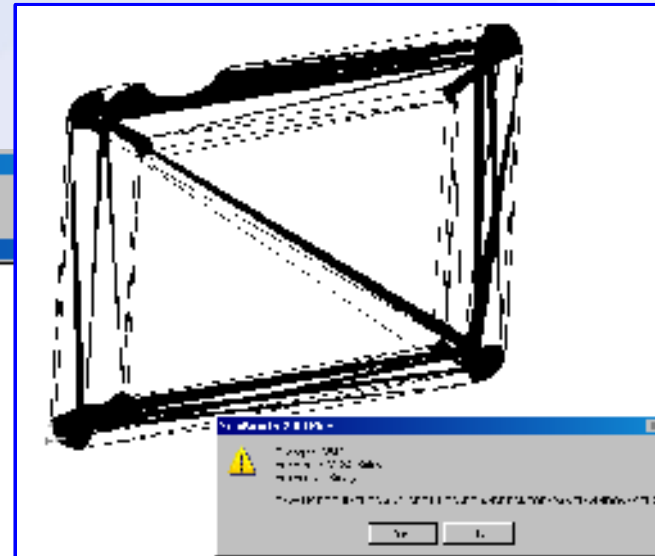
- CAD/CAM Software tool chain (the long part)
 - output modification
 - development suggestions
- Bits
- Materials
- Process
- Limitations & upgrades

Software Toolchain

- Solidworks, Blender, Sketchup*, FreeCAD, etc.
- Design an .stl object (* requires a separate tool chain than I describe here as it doesn't export .stl)

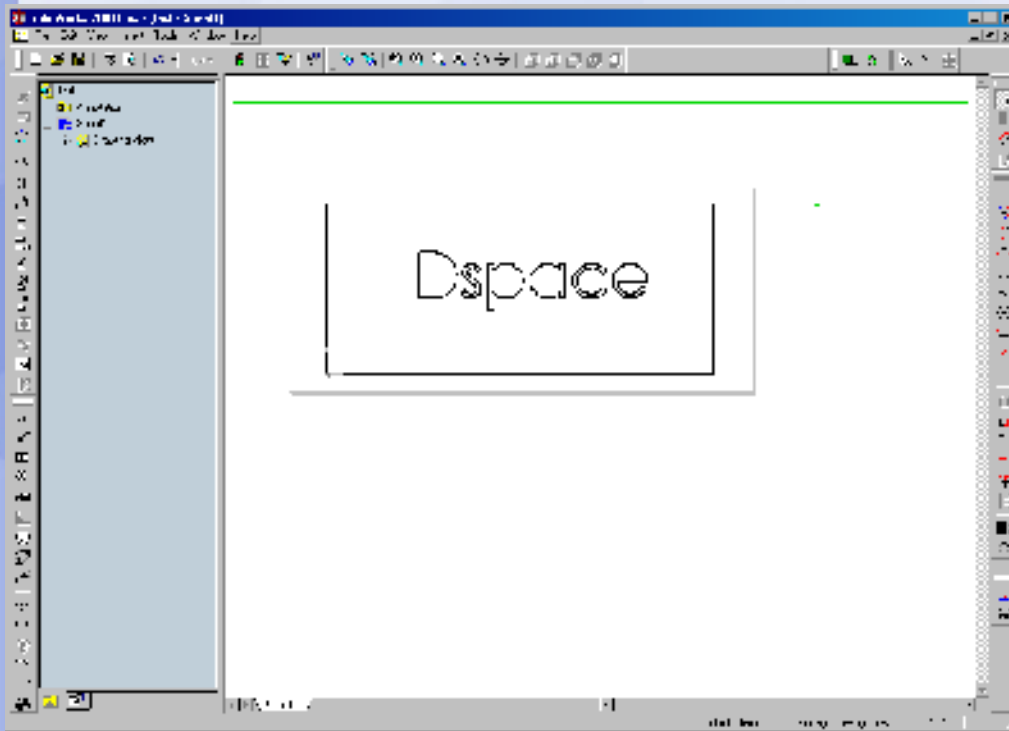


-Base should be on Front plane
-Don't forget case-sensitive .stl vs .STL files for later software.



Software Toolchain

- Solidworks, Blender, Sketchup*, FreeCAD, etc.
- Design an .stl object (* requires a separate tool chain than I describe here as it doesn't export .stl)



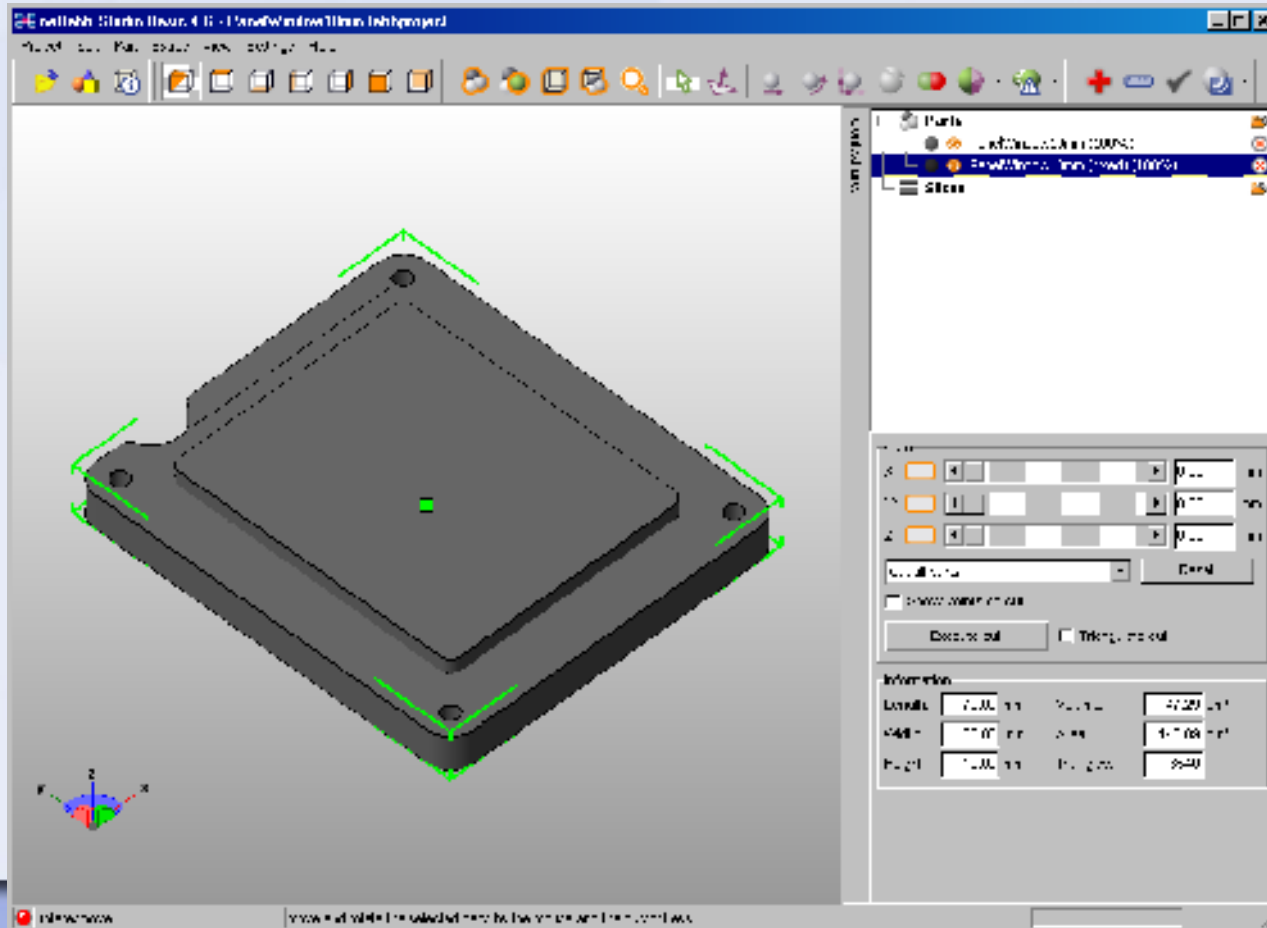
OR, if engraving is your goal,

produce a simplified .dxf (2D) drawing.

-Base should be on Front plane
-Don't forget case-sensitive .dxf vs .DXF files for later software.

Software Toolchain

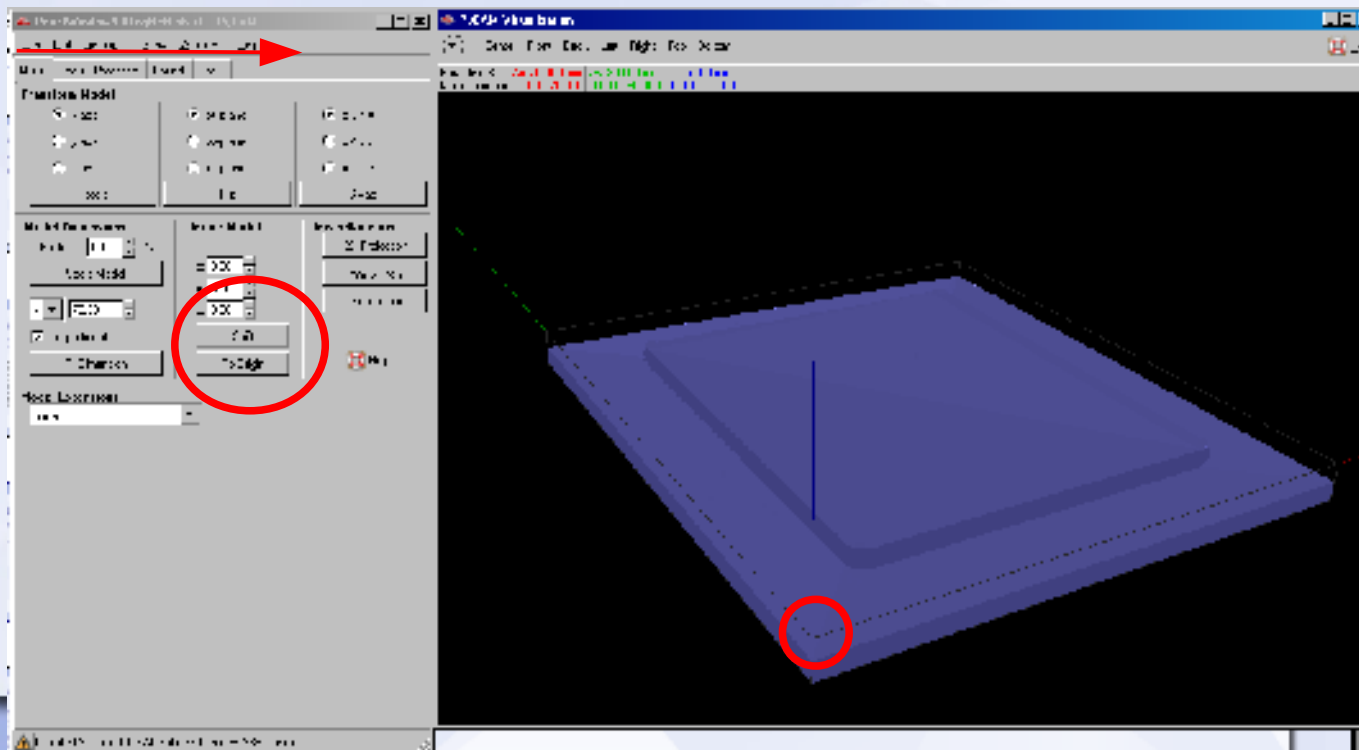
- Solidworks, Blender, Sketchup*, FreeCAD, etc.
- Trim/repair object in Netfabb (optional, yes, you can use free version)
- Design an .stl object (* requires a separate tool chain than I describe here as it doesn't export .stl)
- Very occasionally .stl files contain vertices outside the model boundaries, which will confuse simple X, Y, Z robots



Netfabb also allows you to specify .stl file save in binary or ascii formats and tags appropriately. The next steps can't resolve some CAD tags unambiguously.

Software Toolchain

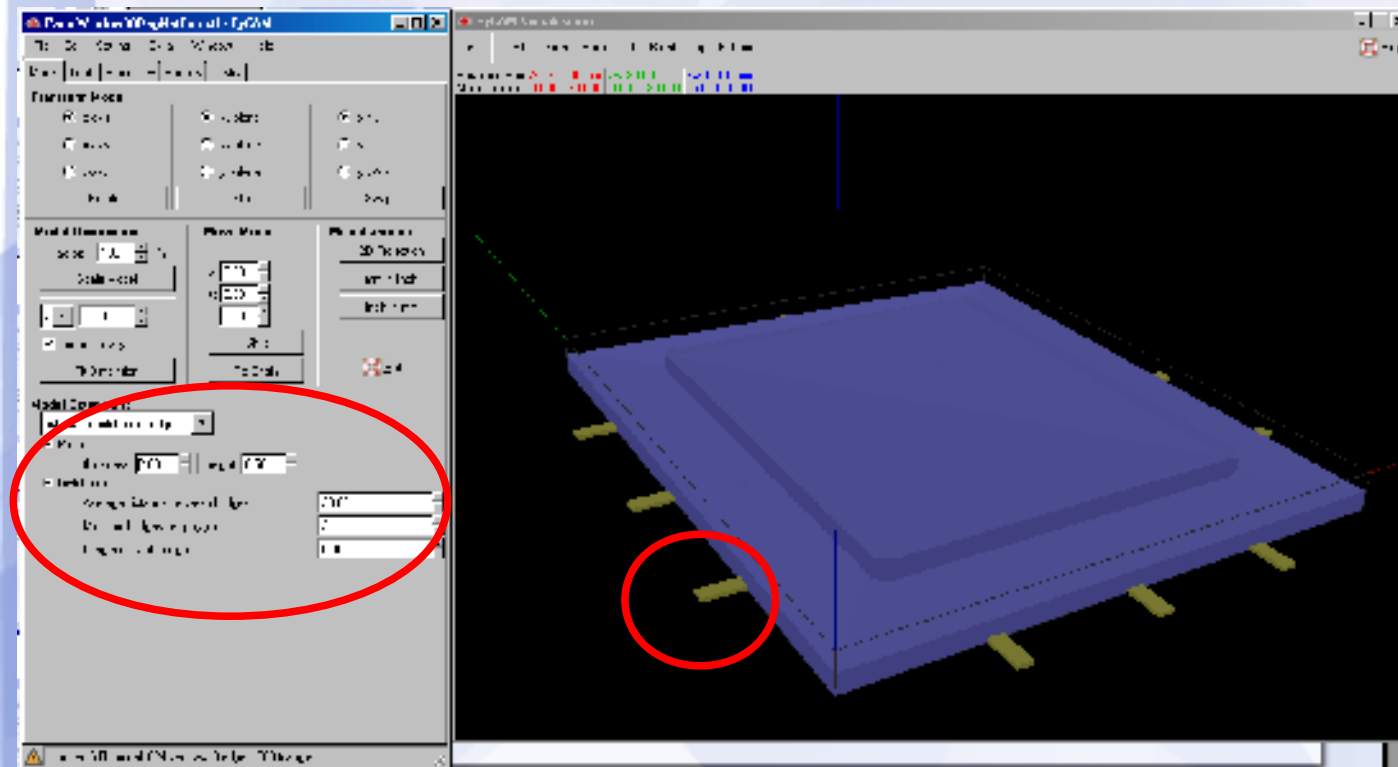
- Solidworks, Blender, Sketchup*, FreeCAD, etc.
- Trim/repair object in Netfabb (optional, yes, you can use free version)
- Slice the object using PyCAM (others available open source or cost, e.g. CAMBam \$125)
- Design an .stl object (* requires a separate tool chain than I describe here as it doesn't export .stl)
- Very occasionally .stl files contain vertices outside the model boundaries, which will confuse simple X, Y, Z robots
- Generates tool paths in finite space as simple text files (g-codes, familiar to Reprap users).



Most of your job decisions have to be made here.

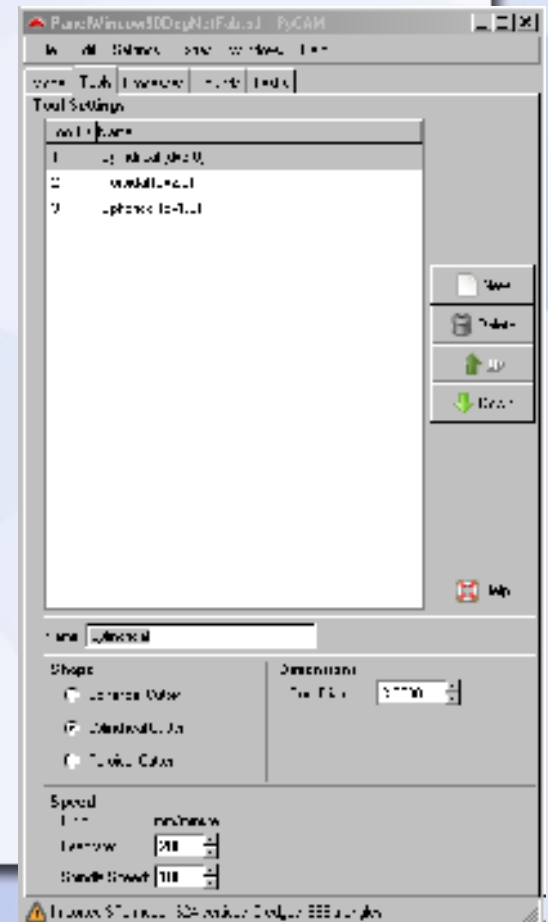
- Define Z=0 (preferably >0mm above model and material), by shifting MODEL relative to Z origin.

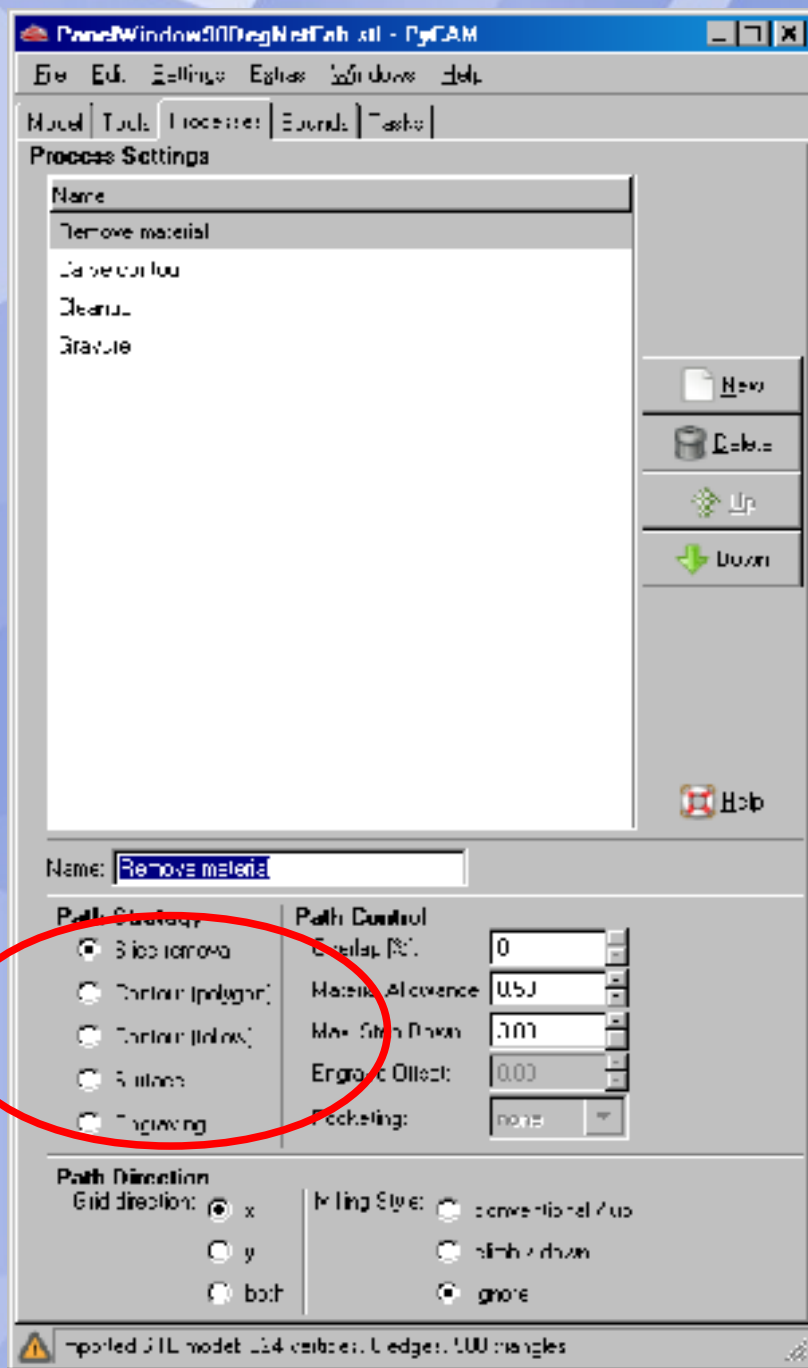
- add bridges (sprue) if you're going to cut through your material completely



- Choose your bit geometry (cylinder, sphere, toroid) and diameter (critical that you verify later, PyCAM sometimes forgets your choice).

- Ignore feed and speed as Arduino won't understand them and you need to modify later.



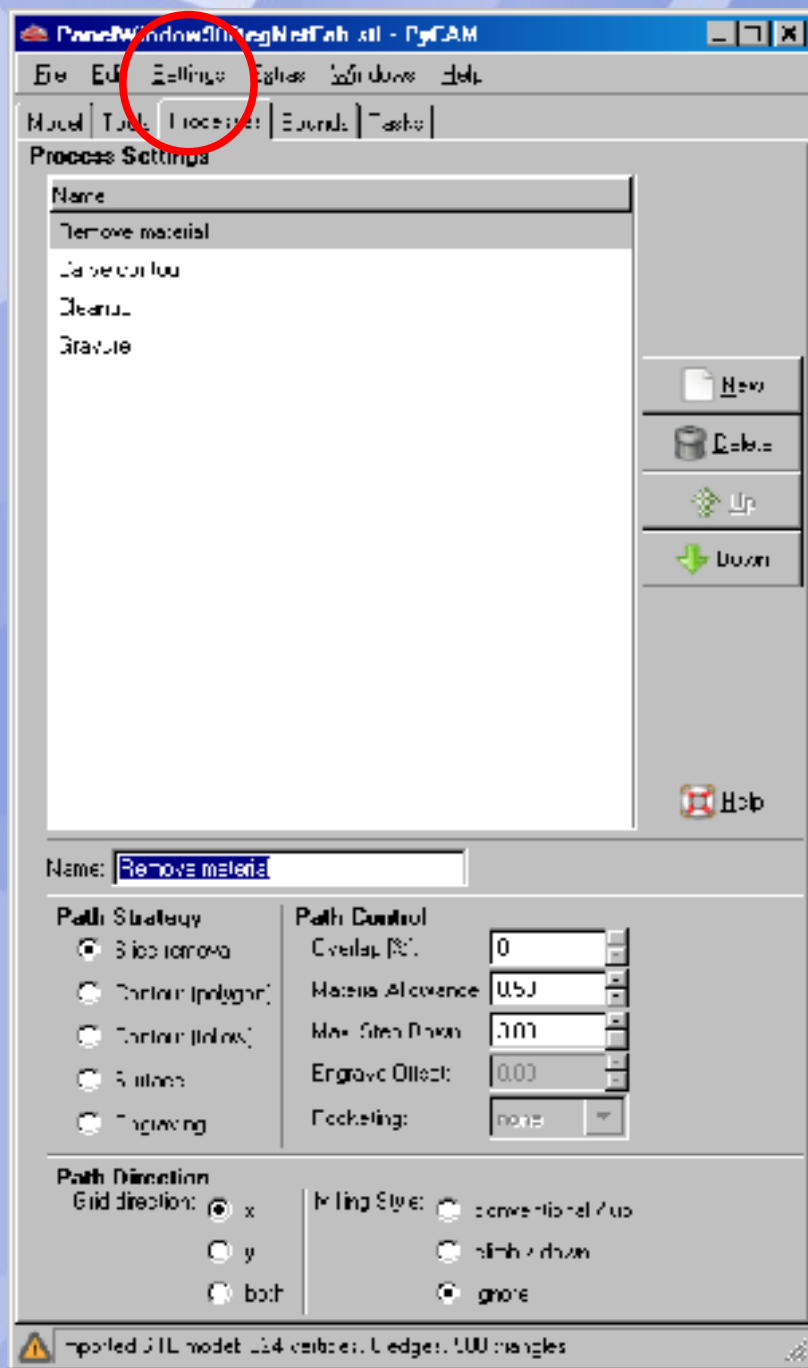


- Define your process. PyCAM is based on the notion that you'll do several passes on any 3D model

- 1st: remove most material
- 2nd: carefully cut contours
- 3rd: surface finish

Slice removal is simple-grid based.
Contour (follow) is detailed cutting (normal)
Surface is plunging.
Engraving is contour (centre of the tool).

Typically you'd want $\leq 50\%$ overlap for most bit and non-powdering material combinations.
CAREFULLY consider max. step down after experimenting on the material/router yourself.



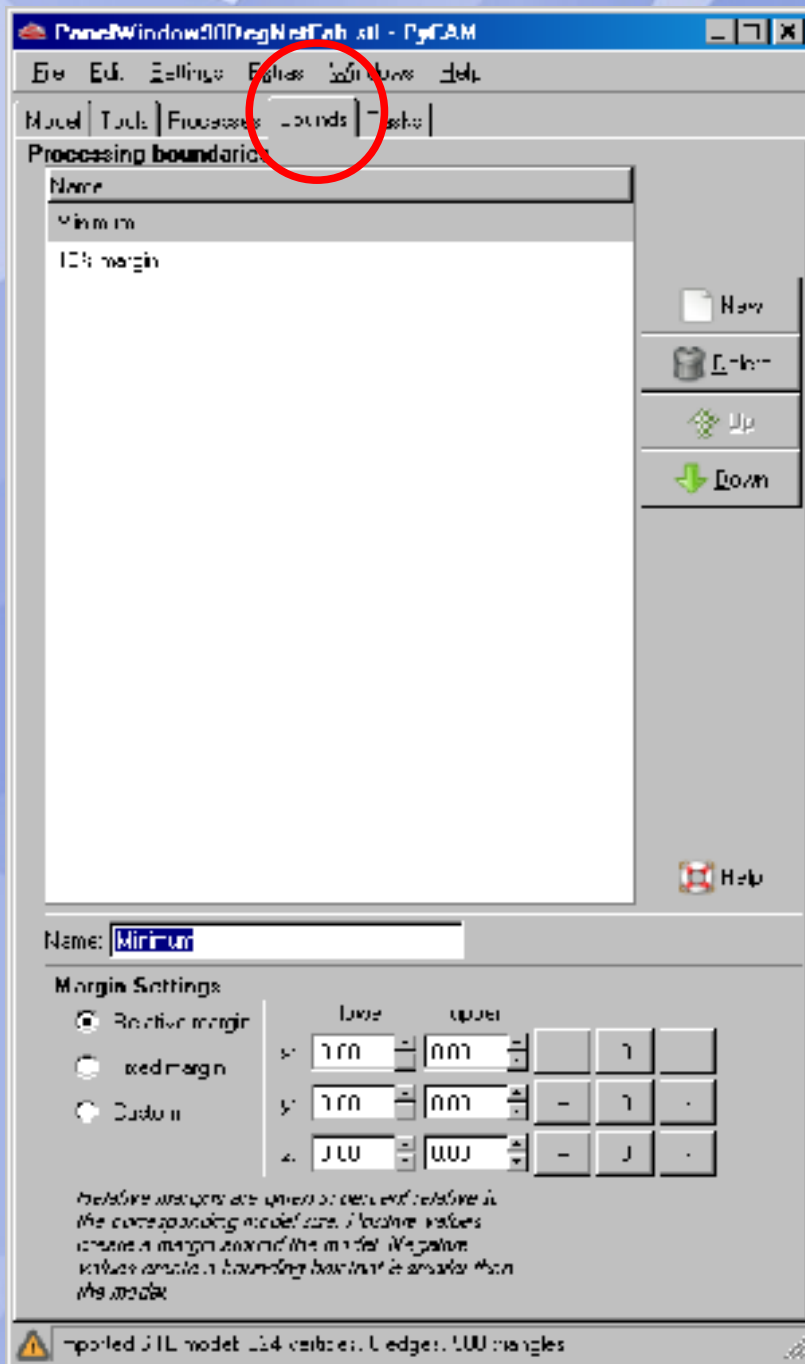
- Define your process. PyCAM is based on the notion that you'll do several passes on any 3D model

- 1st: remove most material
- 2nd: carefully cut contours
- 3rd: surface finish

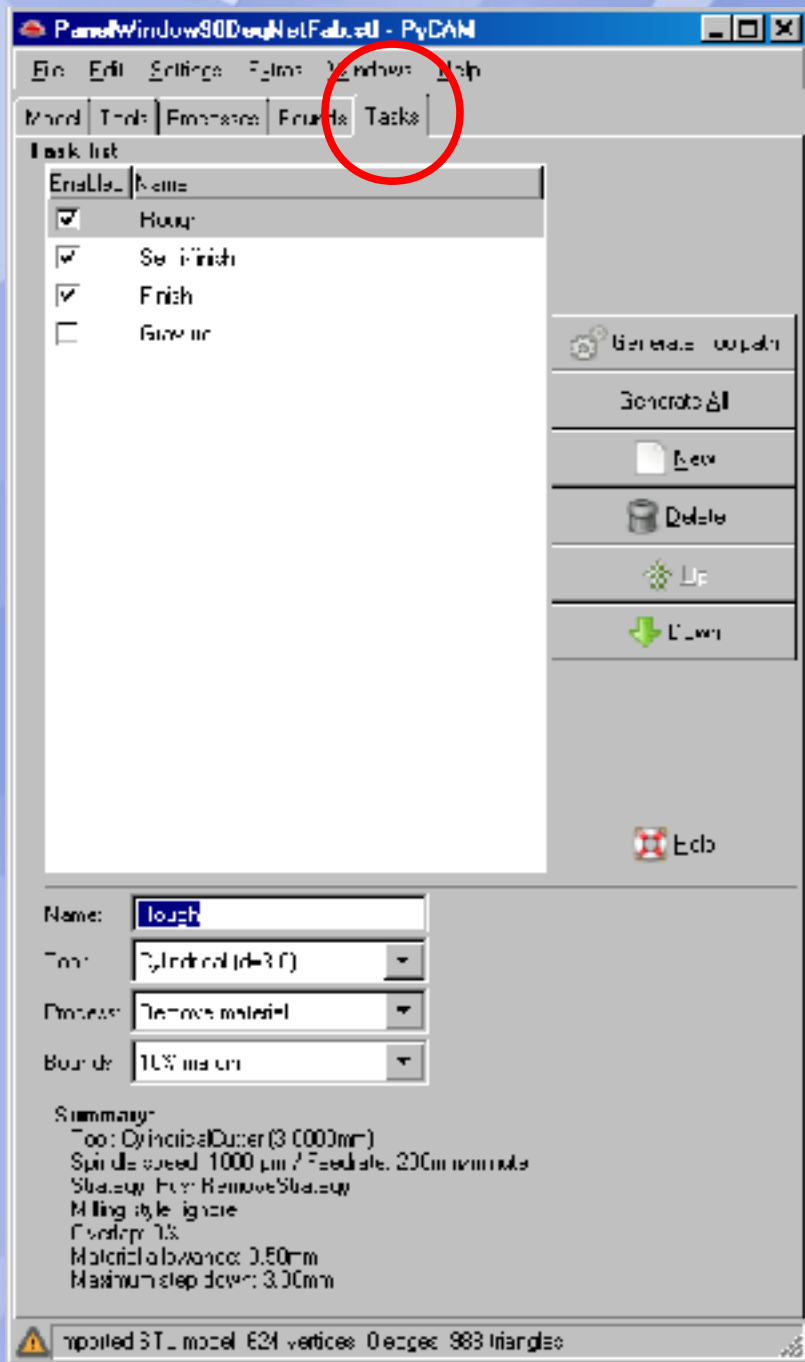
Slice removal is simple-grid based.
Contour (follow) is detailed cutting (normal)
Surface is plunging.
Engraving is contour (centre of the tool).

Typically you'd want $\leq 50\%$ overlap for most bit and non-powdering material combinations. CAREFULLY consider max. step down after experimenting on the material/router yourself.

Incidentally, the most important SETTING is safe-travel Z height at which the tool can travel at top speed unobstructed. Too high a value (e.g. 25 mm) extends job time a lot, too low and you either ruin your job or break stuff.



- Bounds are what you'd expect. Relative margin is relative to the origin.



Tasks Tab

Choose only ONE tool path at a time for the Shapeoko.

Verify your bit and process for the task you choose.

Generate Toolpath. Here is where PyCAM's shortfall becomes apparent. Note that I simplified this model to only 988 triangles from the original model with 6,500. This model takes a minute or so to generate a 50% overlap toolpath in a grid pattern on a 2.8GHz duo-core running Python 2.7 in WinXP. The original model failed after completing 10 of 11 slices after 10 hours of processing.

After your toolpath is generated you absolutely should use the SIMULATE button that pops up so you can see how the tool tip will move relative to the model so that you can prevent problems and ensure that the task is what you really wanted to do (i.e. remember 'follow contour' assumes you've already removed most material).

Software Toolchain

- Solidworks, Blender, Sketchup*, FreeCAD, etc.
- Trim the object in Netfabb (optional, yes, you can use free version)
- Slice the object using PyCAM (others available open source or cost, e.g. CAMBam \$125)
- Simultaneously use a g-code streamer (e.g. Universal G-Code Streamer in Java) which is received by GRBL running on the Arduino
- Design an .stl object (* requires a separate tool chain than I describe here as it doesn't export .stl)
- Very occasionally .stl files contain vertices outside the model boundaries, which will confuse simple X, Y, Z robots
- Generates tool paths in finite space as simple text files (g-codes, familiar to Reprap users).
- You can use UGcS as a straight serial terminal to zero the tool tip and read machine parameters. UGcS sends commands in small chunks to not overload Arduino buffer and receives 'OK' from completed commands to follow job progress.

Except for intentional upgrades, we want GRBL to remain stable on the Arduino so no need to access Shapeoko with the Arduino IDE.

```
*H:\Documents and Settings\ADMIN\Desktop\test.ngc - Notepad++
File Edit Search View Format Language Settings Tools Plugins Window Help
THEFACTE... 2012/03/04 gcode language THEFACTE... 2012/03/04
1 ;PYCAM_PPCAR_PATH: P1: Home: H:\CAM\2012\gcode\12_7027
2 ;PYCAM_PPCAR_PATH: P2: Home: 2012 03 10 18:12:45.686000
3 ;PYCAM_PPCAR_PATH: P3: Home: 1.5.1.1
4 ;Positional machine time: 0 seconds
5 ;Tool change and tool change command line!
6 ;Tool change and tool change command line!
7 ;Tool change and tool change
8 ;54 tool change and tool change
9 ;Tool change and tool change
10 ;Tool change
11 ;Tool change and tool change
12 E200.00000
13 E300.00000
14 ;PYCAM_TOOLPATH_SETTINGS: END
15 ;[Home]
16 g0 x = 0.0
17 g0 y = 125.340566436
18 g0 z = 181.870566100
19 g0 x = 23.7748764105
20 g0 y = 172.340566451
21 g0 z = 0.0
22 ;
23 ;[Tool]
24 g0 x = 0.0
25 g0 y = 177.0
26 g0 z = 20.0
27 g0 x = 1.5
28 g0 y = 200.0
29 ;
30 ;[Panic]
31 g0 x = 0.0
32 g0 y = 177.0
33 g0 z = 20.0
34 g0 x = 1.5
35 g0 y = 200.0
36 g0 z = 0.0
37 g0 x = 0.0
38 g0 y = 177.0
39 g0 z = 20.0
40 ;
41 ;PYCAM_TOOLPATH_SETTINGS: END
42 E1 H6
43 E2 E15.0000
44 E3 retract spindle!
45 E34 28 (last tool 3 seconds!)
46 X155.2105 Y148.4847
47 Z1.20.0000
48 X156.8849 Y144.5630
49
50 X156.1665 Y148.6535
51
```

```
*H:\Documents and Settings\ADMIN\Desktop\test.ngc - Notepad++
File Edit Search View Format Language Settings Tools Plugins Window Help
THEFACTE... 2012/03/04 gcode language THEFACTE... 2012/03/04
1 NO (unsafe approach)
2 G0 Z5.000 (or whatever your safety height is)
3 X155.2105 Y148.4847
4 G1 Z0.0000
5 Auto: 28% Tool: 0.00
6
7 Auto: 28% Tool: 0.00
8
9 Auto: 28% Tool: 0.00
10
11 X156.8849 Y144.5630
12
13 X156.1665 Y148.6535
14
15 X156.8849 Y144.5630
16
17 X156.1665 Y148.6535
18
19 X156.8849 Y144.5630
20
21 X156.1665 Y148.6535
22
23 X156.8849 Y144.5630
24
25 Auto: 28% Tool: 0.00
26
27 Auto: 28% Tool: 0.00
28
29 X156.8849 Y144.5630
30
31 X156.1665 Y148.6535
32
33 X156.8849 Y144.5630
34
35 Auto: 28% Tool: 0.00
36
37 Auto: 28% Tool: 0.00
38
39 X156.8849 Y144.5630
40
41 X156.1665 Y148.6535
42
43 Auto: 28% Tool: 0.00
44
45 Auto: 28% Tool: 0.00
46
47 X156.8849 Y144.5630
48
49 X156.1665 Y148.6535
50
51 X156.8849 Y144.5630
52
```

We all know to expect life to not be easy. PyCAM is a neat utility, but it is in development and not designed specifically for the Shapeoko, so you must edit your g-code (.ngc) text file. It's not as bad as it sounds when you use find/replace.

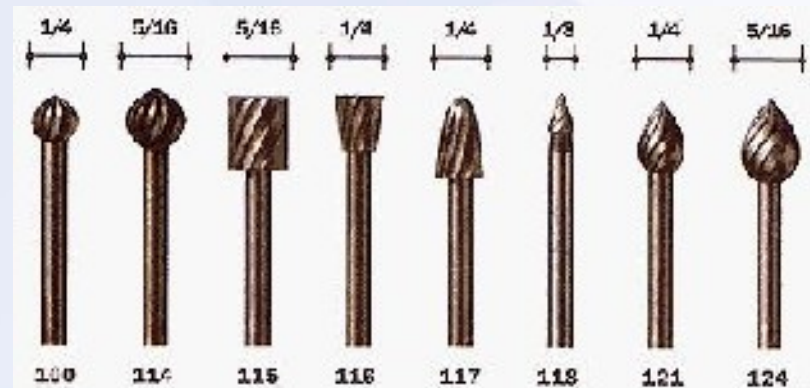
- Remove all commands that confuse and/or fill up the Arduino buffer:
 - 1) anything starting with ;
 - 2) Any commands other than G0,G1,X,Y,Z, P

Add these lines to the end of your gcode.
G0 Z5.000 (or whatever your safety height is)
G1 X0.000
G1 Y0.000

So your tool returns to home ready for the next pass (otherwise you'll lose your place).

Bits

As you'd expect, the choice of bit affects your final product look. Compared to manual use of the router, you must consider that you have FAR less lateral pressure at your disposal so bit sharpness is key (i.e. carbide bits probably better than HSS in most cases). Choose your bits with consideration of your material and the bits ability to plunge.



Materials

Construction material will be foremost in your mind from the very first step. You can modify your spindle speed and tool feed within certain limits, but most “Dremel” bits cut using high speed rather than awesome bit geometries.

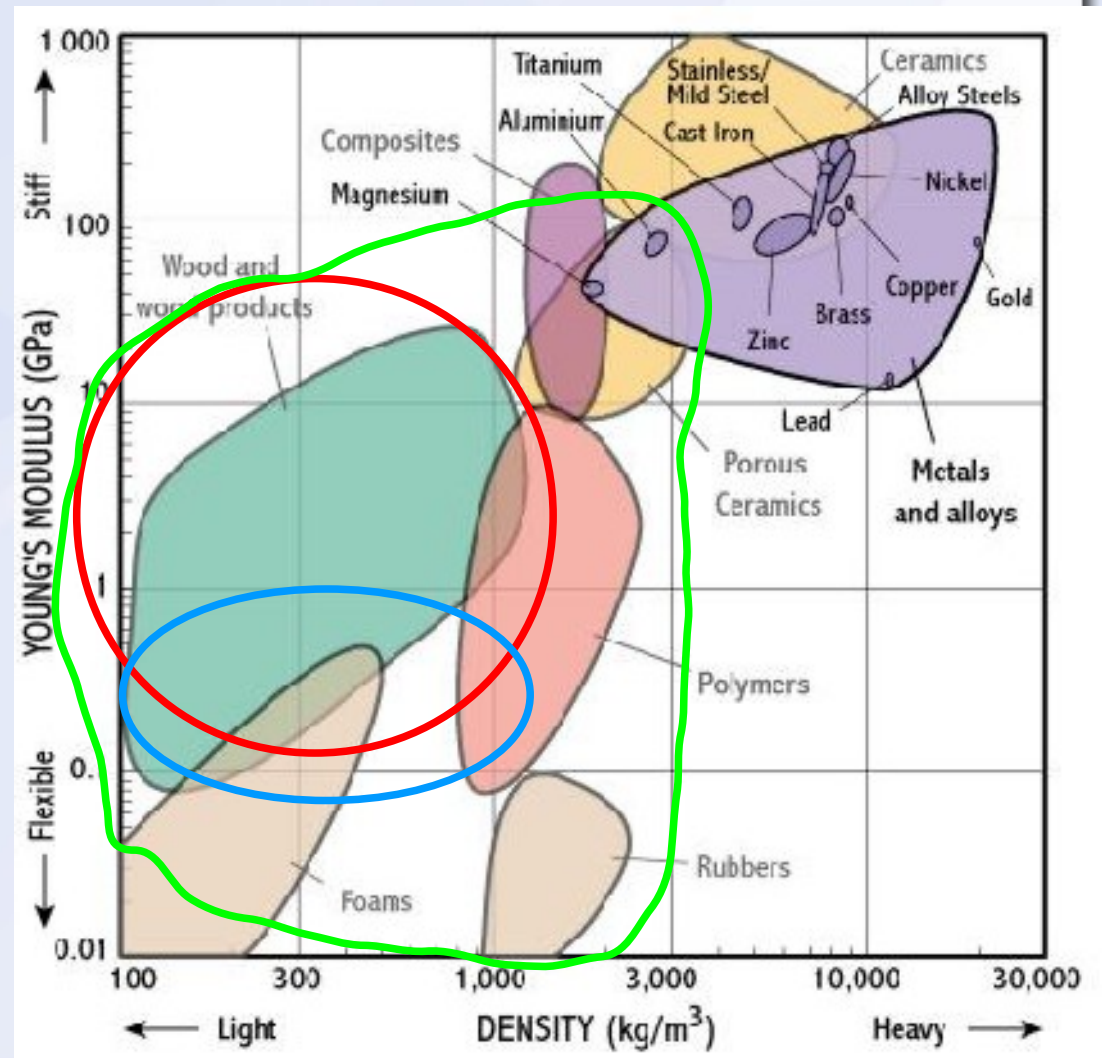
If you're using plastic and you're melting instead of cutting, lower your feed rate. If the cutter is sticking in the groove it's milling, then decrease speed or increase feed.

In my limited experience with the Shapeoko, I suspect the following operational envelopes contrary to advert.

Profile cutting

Drilling

Milling



If it lives here I recommend continuing the material/feed/speed table I've started.

Process with Current Machine

- 1) Mount and level raw material on table, manually move your tool over surface to ensure you don't dig in before powering up electronics.
- 2) Prepare g-code file as discussed
- 3) Start up UGcS
- 4) Connect to comport (4 on this laptop)
- 5) Issue serial command \$ to get current Arduino values, set feed and travel rates as appropriate \$4=100 means feedrate while cutting of 100 mm/minute. \$5=1000 means travel rate of 1000 mm/minute at safe-height.
- 6) Move tool head to appropriate lateral home by issuing X100, Y100, etc. commands, remember values are cartesian from origin NOT steps
- 7) Move tool head to level you decided in PyCAM. I recommend using either locating at 0, sliding paper underneath and step to it in 0.25 mm increments OR (better) step to a soft nylon or acetal surface 3.0 mm above your material.
- 8) Hit the Reset / Zero button on Arduino shield
- 9) Use the file feed pane on UGcS to load your file
- 10) Power on router
- 11) 'Send' file
- 12) Supervise
- 13) Router will complete what's in the buffer even if you hit cancel or pause in the software.

Reset / Zero



Modifications

Some people have made cool things with Shapeoko, but after playing with it myself several modifications come to mind. It turns out that most of the cool stuff was built with machines similarly modified or very careful choice of materials.



Modifications

Some obvious python ones from tool chain biggest being invalid GRBL commands and rehoming on milling tasks, but in terms of hardware:

----- Easy -----

- 1) X,Y home switches (electronics/GRBL ready for them) I brought some in if someone wants to add them.
- 2) Remote Z home switch (easily printed with Reprap)
- 3) Makerslide particle baffle or extraction system
- 4) Right angles to stiffen gantry movements (lag is considerable)
- 5) Level base and improve clamping (v-grooves maybe)
- 6) Make sound-dampening box
- 7) Rewire friggin' crash switch
- 8) Upgrade router

--- Hard ---

- 1) Spring-load idlers for variable tension
- 2) Gear down or upgrade steppers for better torque.
- 3) Swing Z or tool change head

Others?

