Algorithmic Art
~
Converting Math into Motion

Ed Nisley • KE4ZNU
ed.nisley@pobox.com
softsolder.com

ACM - Poughkeepsie Chapter
27 January 2020

Art is making something out of nothing

...

Frank Zappa

https://en.wikipedia.org/wiki/Frank_Zappa

Art is making something out of nothing

and selling it.

Frank Zappa

https://en.wikipedia.org/wiki/Frank_Zappa

Euler’s Identity

$$e^{i\pi} + 1 = 0$$

https://wikipedia.org/wiki/Euler's_identity

The most important thing in art is The Frame.

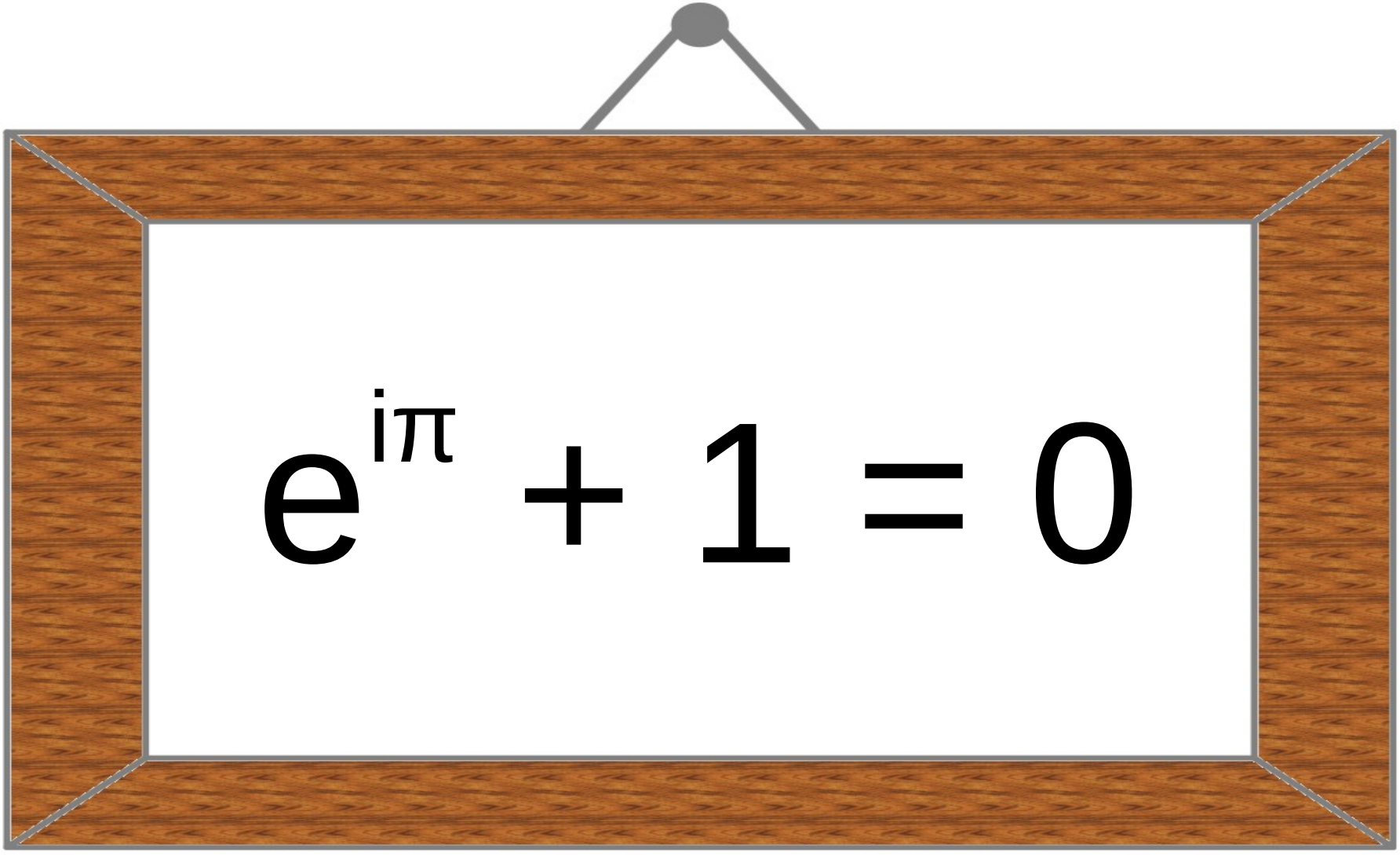
… without this humble appliance, you can't know where The Art stops and The Real World begins.

You have to put a ‘box’ around it because

otherwise, what is that shit on the wall?

Frank Zappa

https://en.wikipedia.org/wiki/Frank_Zappa



https://wikipedia.org/wiki/Euler's_identity

Algorithm

… a finite sequence of well-defined, computer-implementable instructions,

typically to solve a class of problems or to perform a computation.

https://en.wikipedia.org/wiki/Algorithm

Algorithmic art is … visual art,
… generated by an algorithm.

Algorithmic artists are sometimes called

algorists.

https://en.wikipedia.org/wiki/Algorithmic_art

Algorithm?

$$A \cdot \sin(\omega t + \phi)$$

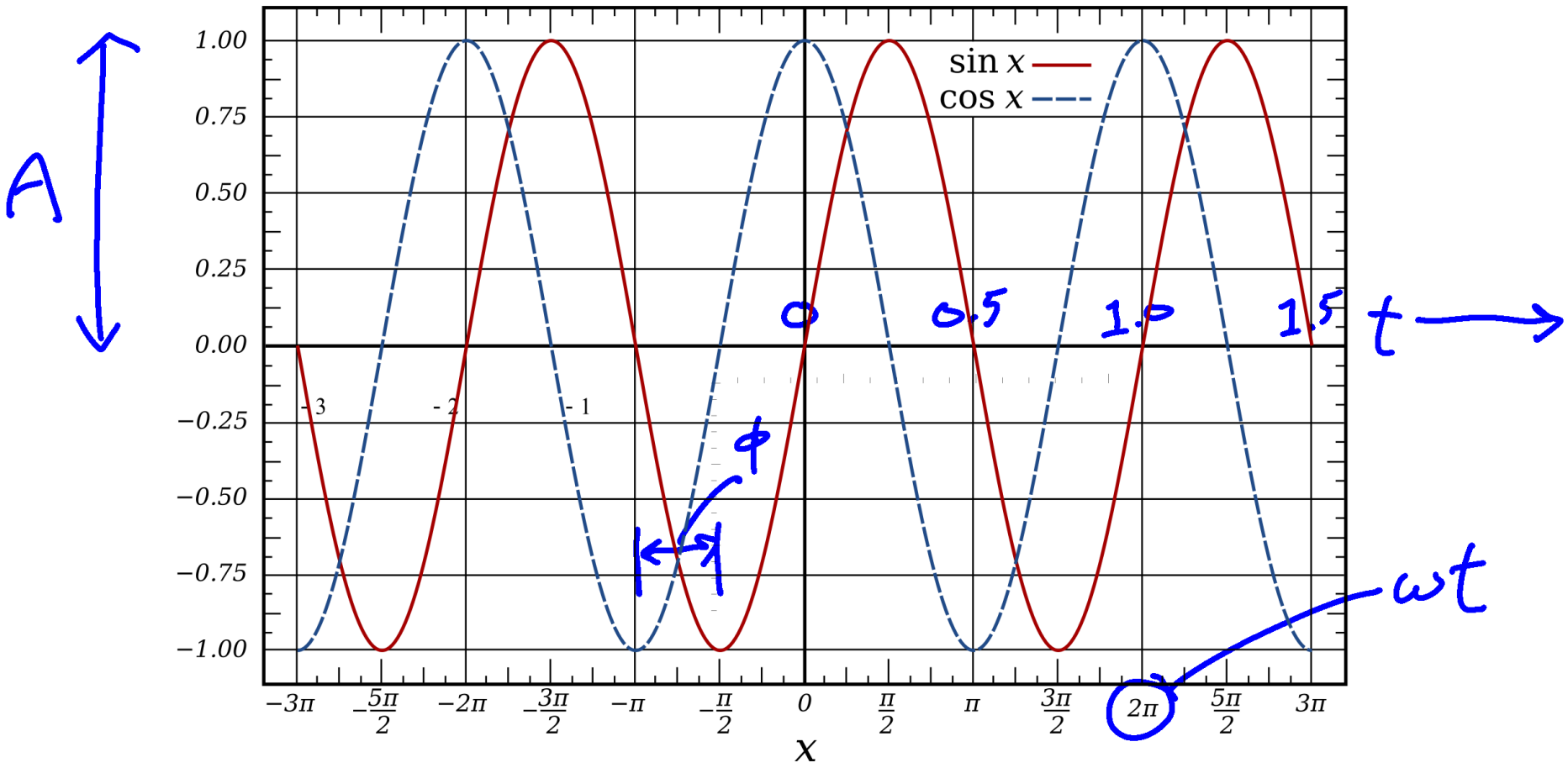
https://en.wikipedia.org/wiki/Sine_wave

Algorithm!

$$A \cdot \sin(\omega t + \phi)$$

https://en.wikipedia.org/wiki/Sine_wave

$A \cdot \sin(\omega t + \phi)$



https://en.wikipedia.org/wiki/Sine_wave

Algorithm!

```
for (int i=0; i < strip.numPixels(); i++) {      // for each pixel
  byte Value[PIXELSIZE];

  for (byte c=0; c < PIXELSIZE; c++) {           //  ... for each color

    Value[c] = (255.0 / 2.0) *
               (1.0 + sin(Pixels[c].Step * Pixels[c].StepSize - i*Pixels[c].Phase));
  }

  byte WhiteBias = min(min(Value[RED],Value[GREEN]),Value[BLUE]);

  uint32_t UniColor = strip.Color((Value[RED] - WhiteBias) * Pixels[RED].MaxPWM/255,
                          (Value[GREEN] - WhiteBias) * Pixels[GREEN].MaxPWM/255,
                          (Value[BLUE] - WhiteBias) * Pixels[BLUE].MaxPWM/255,
                          WhiteBias * Pixels[WHITE].MaxPWM/255);
  strip.setPixelColor(i,UniColor);
}
```

https://softsolder.com/2016/10/20/hp-7475a-plotter-coordinate-pruning/

$A \cdot \sin(\omega t + \phi) + B$

```
for (int i=0; i < strip.numPixels(); i++) {      // for each pixel
  byte Value[PIXELSIZE];

  for (byte c=0; c < PIXELSIZE; c++) {           //  ... for each color

    Value[c] = (255.0 / 2.0) *
               (1.0 + sin(Pixels[c].Step * Pixels[c].StepSize - i*Pixels[c].Phase));
  }
  byte WhiteBias = min(min(Value[RED],Value[GREEN]),Value[BLUE]);

  uint32_t UniColor = strip.Color((Value[RED] - WhiteBias) * Pixels[RED].MaxPWM/255,
                                  (Value[GREEN] - WhiteBias) * Pixels[GREEN].MaxPWM/255,
                                  (Value[BLUE] - WhiteBias) * Pixels[BLUE].MaxPWM/255,
                                  WhiteBias * Pixels[WHITE].MaxPWM/255);
  strip.setPixelColor(i,UniColor);
}
```

R/G/B

B

t

ω

A

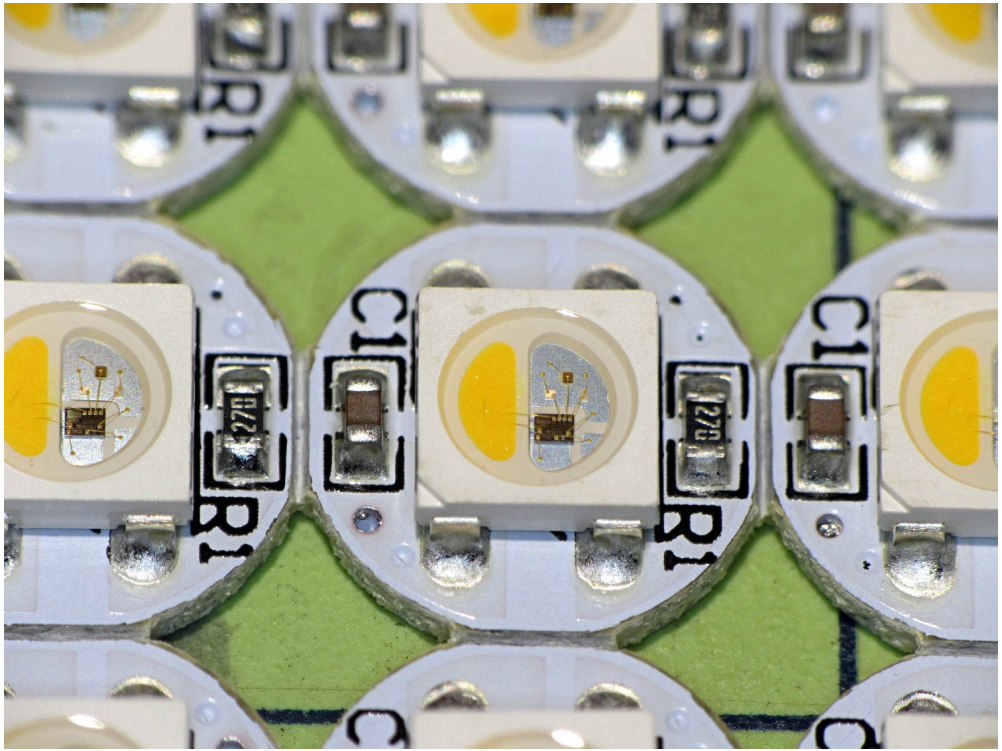
φ

$A \cdot (B + \sin(\omega t + \phi))$

https://softsolder.com/2016/10/20/hp-7475a-plotter-coordinate-pruning/

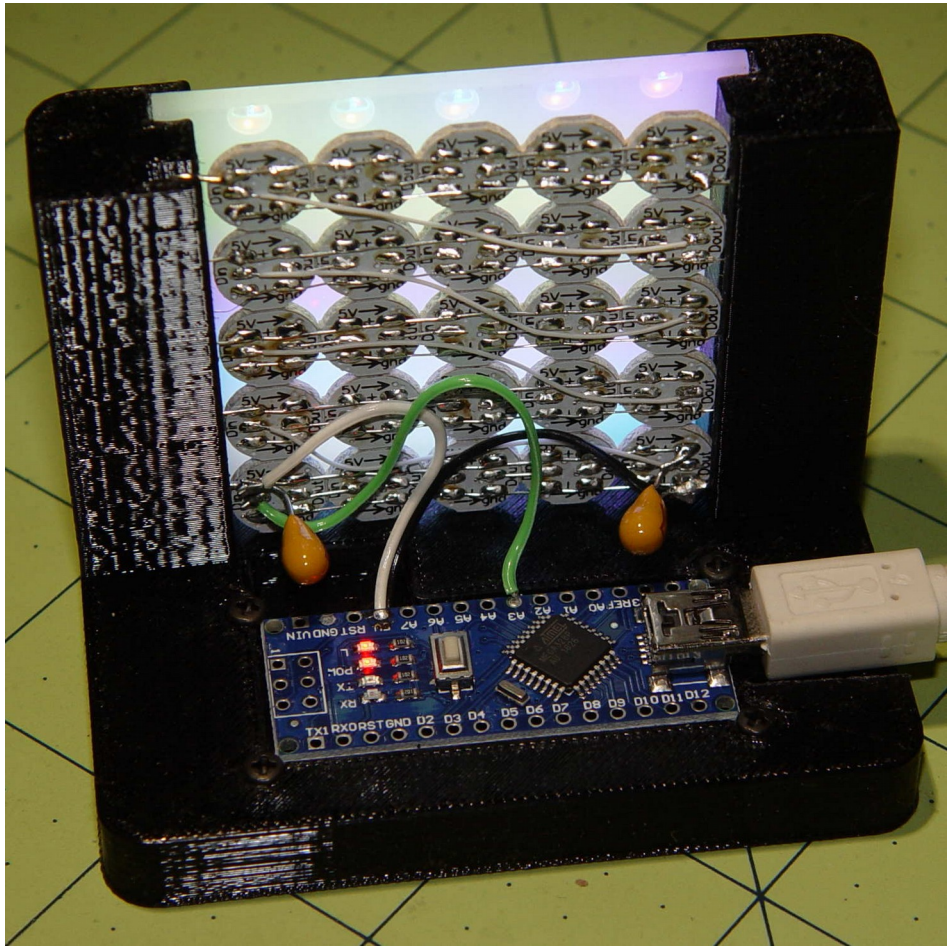
SK6812 – RGBW LEDs

- Digital color
  - Red Green Blue White
  - Range = 0 to 255
- Serial data stream
  - 32 bits = 8 bits × 4
  - Self-clocking PWM
  - Arduino library
- Seems easy enough



https://softsolder.com/2017/03/14/sk2812-rgbw-led-test-fixture/

Arduino Nano + SK6812 LEDs

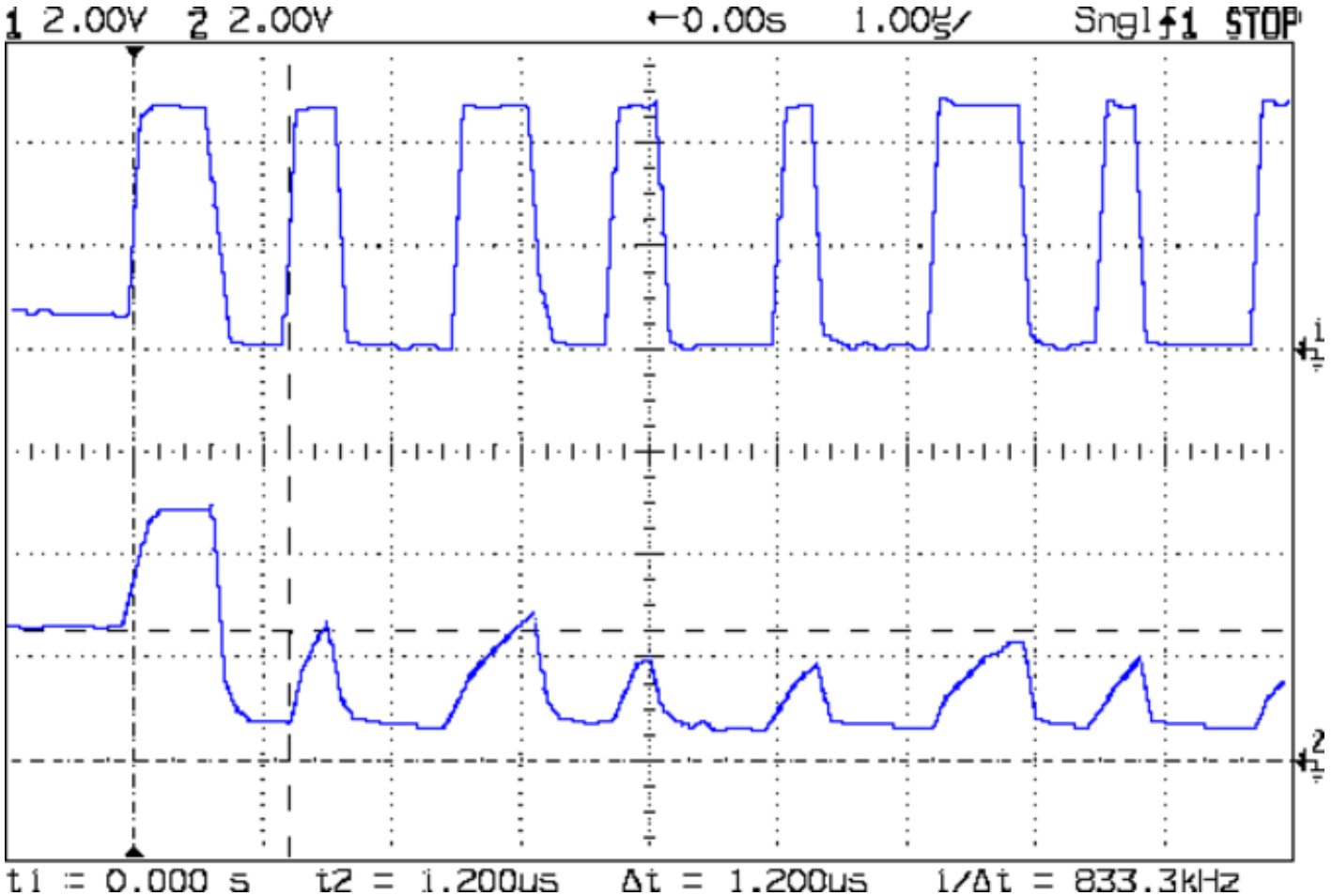


https://softsolder.com/2017/03/14/sk2812-rgbw-led-test-fixture/

Digital Color

https://softsolder.com/2016/07/27/vacuum-tube-leds-bowl-of-fire-floodlight/

Reality Would Like a Word With You



https://softsolder.com/2017/03/21/cheap-ws2812-leds-failure-waveforms/

https://softsolder.com/2016/10/12/vacuum-tube-leds-now-with-morse-code/

Gielis Superformula

$$r = f(\Theta) = \left(\left| \frac{1}{a} \cdot \cos\left(\frac{m}{4} \cdot \Theta \right) \right|^{n_2} + \left| \frac{1}{b} \cdot \sin\left(\frac{m}{4} \cdot \Theta \right) \right|^{n_3} \right)^{\frac{-1}{n_1}}$$

Θ = Theta or Φ = Phi

… whatever …

It's just an angle

https://en.wikipedia.org/wiki/Superformula

Algorithm: One Point

```
def superformula(a, b, m, n1, n2, n3, phi):

    t1 = cos(m * phi / 4.0) / a
    t1 = abs(t1)
    t1 = pow(t1, n2)

    t2 = sin(m * phi / 4.0) / b
    t2 = abs(t2)
    t2 = pow(t2, n3)

    t3 = -1 / float(n1)
    r = pow(t1 + t2, t3)
    if abs(r) == 0:
        return (0,0)
    else:
        return (r * cos(phi), r * sin(phi))
```

Θ = Theta or Φ = Phi
… whatever …
It's just an angle

sites.music.columbia.edu/cmc/chiplotle/manual/chapters/api/tools.html?#chiplotle.tools.mathtools.superformula

Compute. All. The. Points.

```
def supershape( … ):      # excerpted for pedagogic clarity

  point_count = 10 * 1000

  travel = 10 * 2 * pi

  phis = [i * travel / point_count
          for i in range(1 + point_count)]

  points = [tools.mathtools.superformula(a, b, m, n1, n2, n3, phi)
            for phi in phis]
```

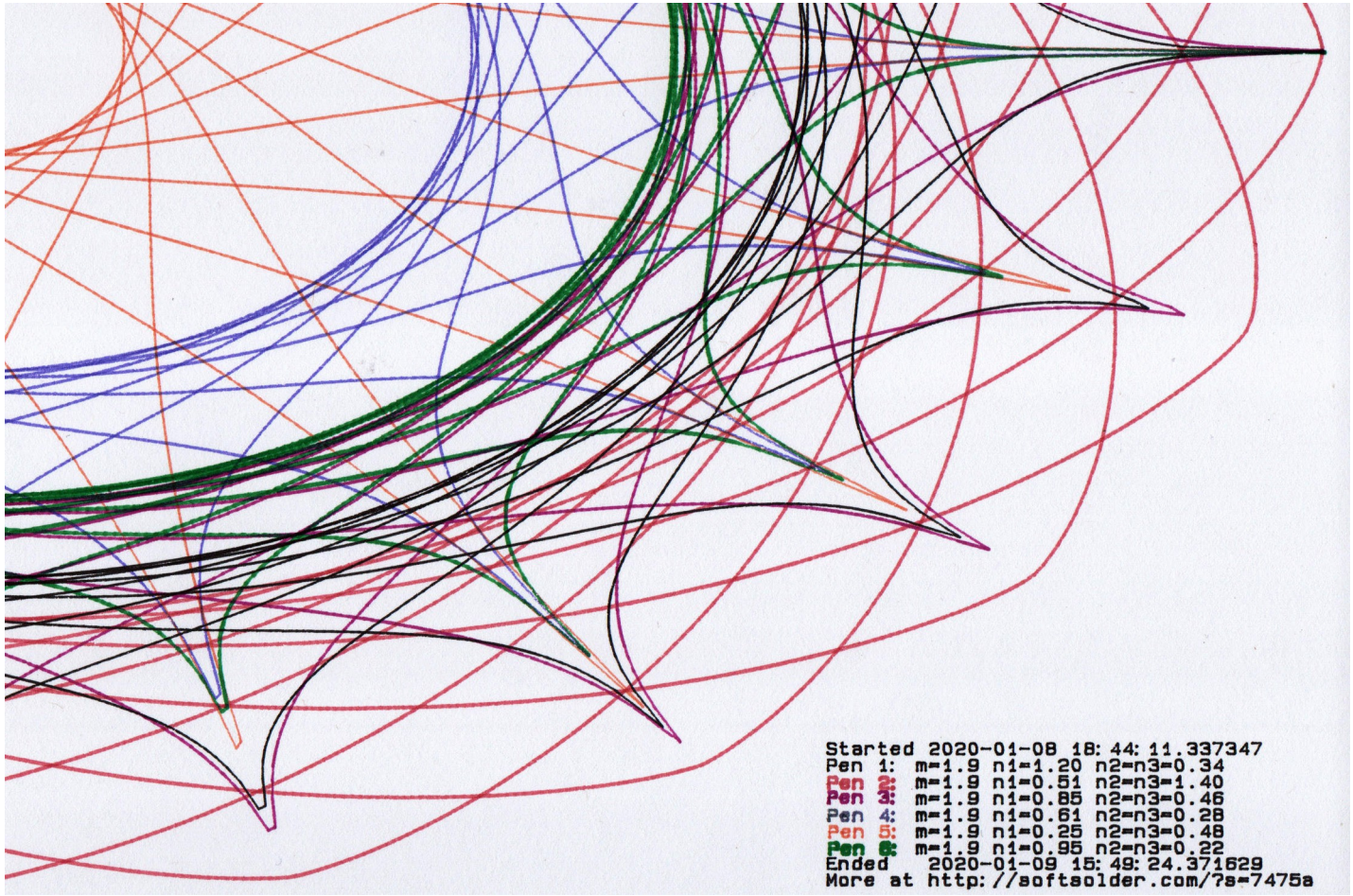
http://sites.music.columbia.edu/cmc/chiplotle/manual/chapters/api/geometry.html?#chiplotle.geometry.shapes.supershape

Algorithm!

```
pen = 1
for n1, n2 in zip(n1_list, n2_list):
  n3 = n2
  print "{0} - m: {1:.1f}, n1: {2:.2f}, n2=n3: {3:.2f}".format(pen, m, n1, n2)
  plt.select_pen(pen)
  plt.write(hpgl.PA([(legendx, legendy - 100 * pen)]))
  plt.write(hpgl.LB("Pen {0}: ".format(pen)))
  plt.select_pen(1)
  plt.write(hpgl.LB("m={0:.1f} n1={1:.2f} n2=n3={2:.2f}".format(m, n1, n2)))

  plt.select_pen(pen)

  e = supershape(maxplotx, maxploty, m, n1, n2, n3)
  plt.write(e)

  pen = pen + 1 if (pen % numpens) else 1
```

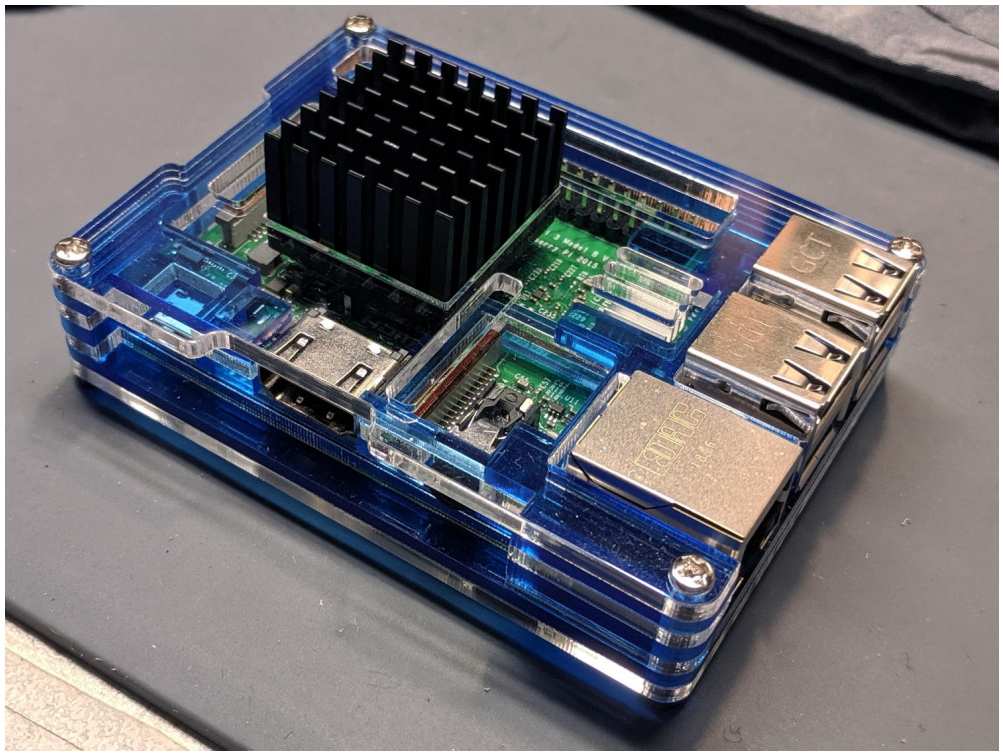
https://softsolder.com/2016/10/20/hp-7475a-plotter-coordinate-pruning/

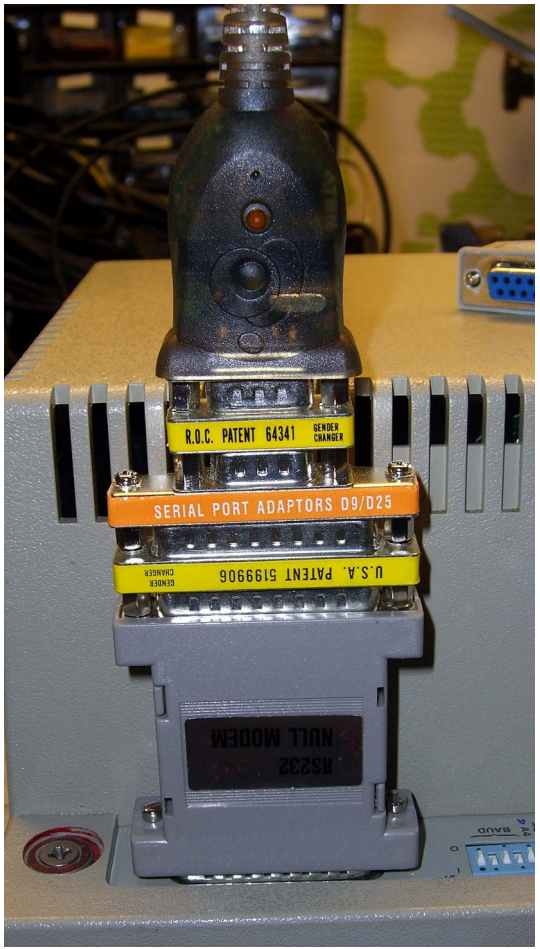
Algorithmic Art!



https://softsolder.com/2016/10/20/hp-7475a-plotter-coordinate-pruning/

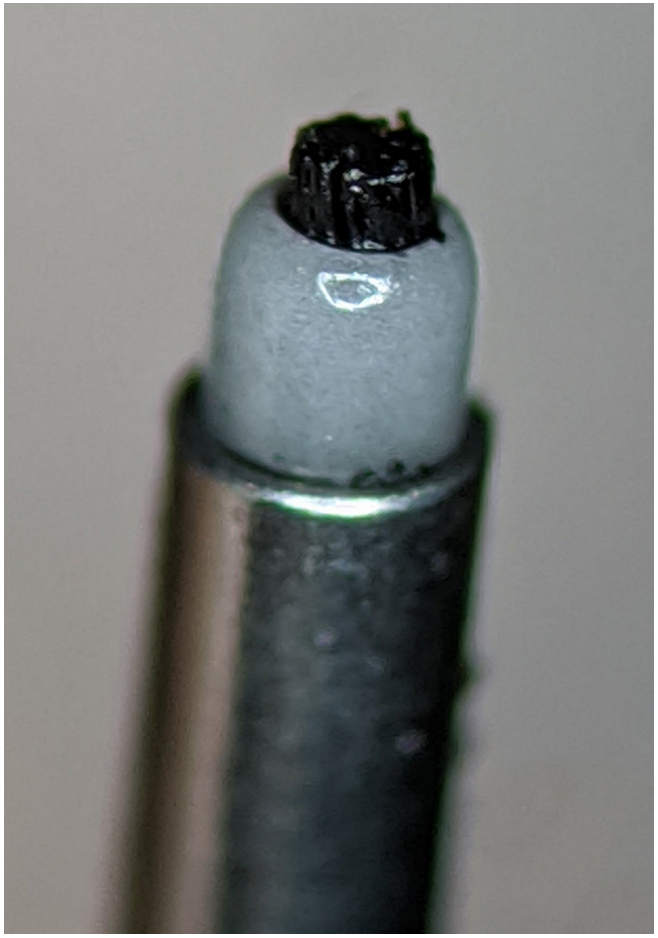
Raspberry Pi + USB Serial Port

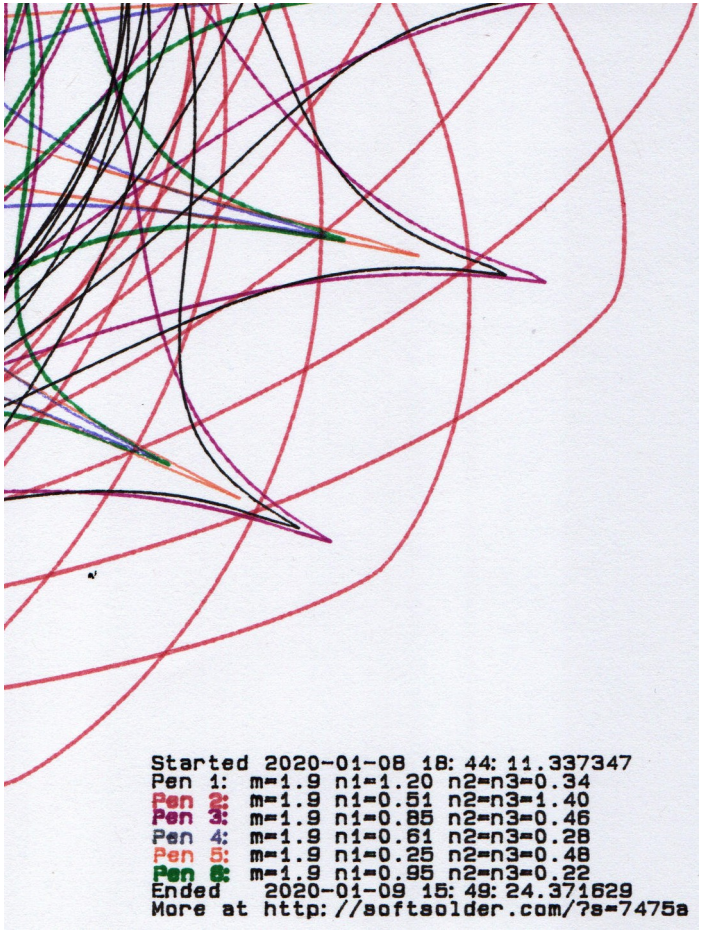




https://softsolder.com/2015/04/20/hp-7475a-plotter-rehabilitation/

Reality Would Like A Word With You





https://softsolder.com/2016/10/20/hp-7475a-plotter-coordinate-pruning/

Almost … All. The. Points.

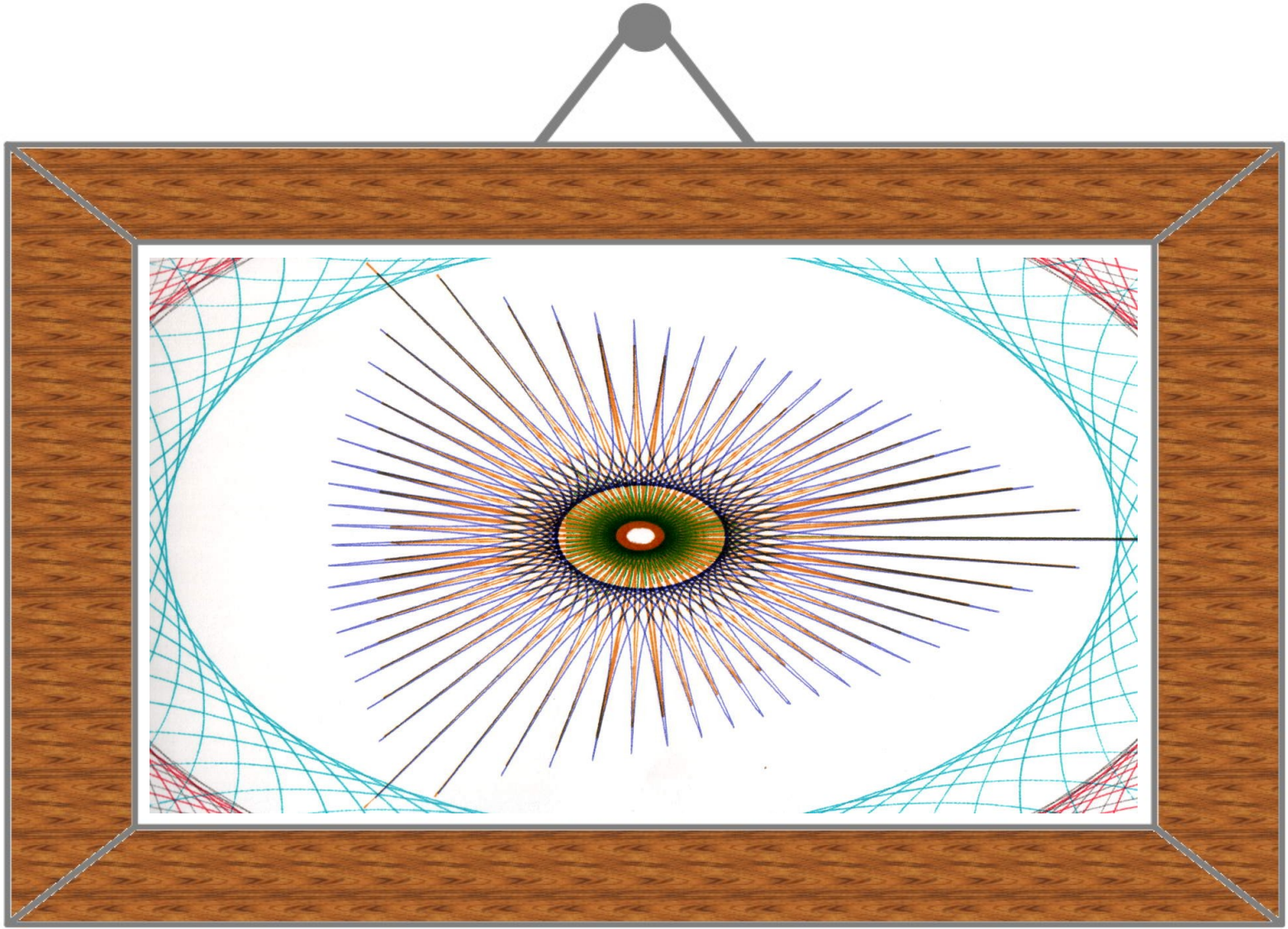
```
points = [tools.mathtools.superformula(a, b, m, n1, n2, n3, x) for x in phis]

path = []
xp, yp = width * points[0][0], height * points[0][1]
path.append(Coordinate(xp, yp))

for pt in points[1:]:
  x,y = width * pt[0], height * pt[1]

  dist = sqrt(pow(x - xp,2) + pow(y – yp,2))

  if dist > 30 :    # 40 plotter units / mmm
    path.append(Coordinate(x, y))
    xp, yp = x, y

path.append(Coordinate(width * points[-1][0], height * points[-1][1]))
print "  Pruned",len(points),"to",len(path),"points"

return Path(path)
```

https://softsolder.com/2016/10/20/hp-7475a-plotter-coordinate-pruning/

Pruned & Plotted Path

```
Instantiated plotter HP7475A in port /tmp/ttyv0:
   Drawing limits: (left 0; bottom 0; right 16640; top 10365)
   Buffer Size: 512
   Max: (8320,5182)
1 - m: 1.9, n1: 0.71, n2=n3: 0.26
  Pruned 10001 to 4582 points
2 - m: 1.9, n1: 0.31, n2=n3: 1.20
  Pruned 10001 to 5724 points
3 - m: 1.9, n1: 0.47, n2=n3: 0.58
  Pruned 10001 to 4661 points
4 - m: 1.9, n1: 0.59, n2=n3: 1.70
  Pruned 10001 to 7383 points
5 - m: 1.9, n1: 0.65, n2=n3: 0.44
  Pruned 10001 to 4932 points
6 - m: 1.9, n1: 1.40, n2=n3: 0.20
  Pruned 10001 to 5233 points
```

https://softsolder.com/2016/10/20/hp-7475a-plotter-coordinate-pruning/



http://softsolder.com/2015/11/02/hp-7475a-plotter-one-button-demo-madness/

Why A Plotter?

The 7475 is engineered to be especially useful in the areas of business graphics, statistics, medicine, numerical control, surveying, and engineering design. With an optional overhead transparency kit, you can produce high quality graphic transparencies from your plotting programs. For faster comprehension, you can present economic trends, engineering or scientific data, marketing plans, profit data, or sales forecasts pictorially. And with this choice of media, you can create paper hardcopy for an individual's attention or transparencies for group presentations.

Whether you tabulate, measure, or compute data, depend on the reliable 7475 to prepare multicolored plots of excellent line quality and high resolution.

https://archive.org/details/HP7475AOperationManual

Hewlett Packard *Graphics* Language

HP-GL

...

was introduced with the plotter HP-8972

in 1977

and became a standard for almost all plotters.

https://en.wikipedia.org/wiki/HP-GL

Programming Languages: 1970s

1976	Mesa	Xerox PARC	ALGOL
1976	SAM76	Claude A.R. Kagan	LISP, TRAC
1976	Ratfor	Brian Kernighan	C, FORTRAN
1976	S	John Chambers at Bell Labs	APL, PPL, Scheme
1976	SAS	SAS Institute	
1976	Integer BASIC	Steve Wozniak	BASIC
1977	FP	John Backus	none (unique language)
1977	Bourne Shell (*sh*)	Stephen R. Bourne	none (unique language)
1977	Commodore BASIC	Jack Tramiel	BASIC
1977	IDL	David Stern of Research Systems Inc	Fortran
1977	Standard MUMPS		MUMPS
1977	Icon (concept)	Ralph Griswold	SNOBOL
1977	Red	Benjamin M. Brosgol *et al.* at Intermetrics for US Dept of Defense	ALGOL 68, CS-4
1977	Blue	John B. Goodenough[7][8] *et al.* at SofTech for US Dept of Defense	ALGOL 68
1977	Yellow	Jay Spitzen *et al.* at SRI International for US Dept of Defense	ALGOL 68
1977	Euclid	Butler Lampson at Xerox Parc, Ric Holt and James Cordy at University of Toronto	
1977	Applesoft BASIC	Marc McDonald and Ric Weiland	BASIC
1978	RAPT	Pat Ambler and Robin Popplestone	APT
1978	C shell	Bill Joy	C
1978	RPG III	IBM	FARGO, RPG, RPG II
1978	HAL/S	designed by Intermetrics for NASA	XPL
1978	Applesoft II BASIC	Marc McDonald and Ric Weiland	Applesoft BASIC
1975	Irvine Dataflow (implementation)	Arvind and Gostelow, University of California, Irvine	

https://en.wikipedia.org/wiki/Timeline_of_programming_languages

Graphic Language: Select

The Select Pen Instruction, SP

DESCRIPTION The select pen instruction, SP, selects and/or stores a pen.

USES The instruction is used to load a pen into the pen holder so that drawing will occur. It can be used to select a pen of a different color or width, during the plotting program. It can be used with a zero parameter or no parameter to store the pen currently in the pen holder into its stall at the end of a program.

SYNTAX

SP pen number terminator

or

SP terminator

EXPLANATION The pen parameter must be in the range of $0 <= n <= 6$. Decimal fractions are truncated. A zero parameter or no parameter stores the pen unless the pen carousel is full. If the pen carousel is full, the plotter will try to put the pen away in the appropriate stall. If the stall is occupied, the plotter will attempt to store the pen in pen stalls 1 through 6 in order. If all the stalls are full, the pen holder will return to its previous location. When a pen parameter is out of range, the parameter is ignored and the pen does not change. If the pen designated for selection is not in its stall, the plotter will attempt to select a pen beginning in stall 1 and continuing through stall 6 until a pen is found.

https://archive.org/details/HP7475AOperationManual

Graphic Language: Pen Up / Down

The Pen Instructions, PU and PD

DESCRIPTION The pen up instruction, PU, and the pen down instruction, PD, raise and lower the pen.

USES The instructions are used to raise and lower the pen during plotting. They may be used with parameters to plot or move to the points specified by the parameters.

SYNTAX

PU terminator
or
PD terminator

and

PU X,Y(, . . .) terminator
or
PD X,Y(, . . .) terminator

https://archive.org/details/HP7475AOperationManual

Graphic Language: Plot!

The Plot Absolute Instruction, PA

DESCRIPTION The plot absolute instruction, PA, moves the pen to the point(s) specified by the X- and Y-coordinate parameters.

USES The instruction can be used together with PD to draw lines or with PU to move the pen to a specific point on the plot. The instruction can be executed without parameters to establish absolute plotting, as opposed to relative plotting for PU or PD instructions with parameters. In this case, the parameters of PU and PD are interpreted as absolute X,Y coordinates until any PR instruction is received.

SYNTAX

PA X_1 coordinate,Y_1 coordinate (,X_2 coordinate, Y_2 coordinate, . . . ,X_n coordinate, Y_n coordinate) terminator

or

PA terminator

https://archive.org/details/HP7475AOperationManual

Graphic Language: Speed Kills

The Velocity Select Instruction, VS

DESCRIPTION The velocity select instruction, VS, specifies the pen-down speed for plotting and labeling operations.

USES The instruction is used to set velocity to a speed other than the default velocity of 38.1 cm/s and to change the acceleration from its default value of 2 g (980 cm/s^2). This instruction should be used to slow velocity to 10 cm/s when plotting on transparency film. A slightly thicker line can be created by slowing down the pen speed on any medium. A pen nearing the end of its life will write with a clearer, sharper, more solid line if the velocity is slowed.

SYNTAX

VS pen velocity terminator
or
VS terminator

https://archive.org/details/HP7475AOperationManual

Domain Specific Language

The Shade Wedge Instruction, WG

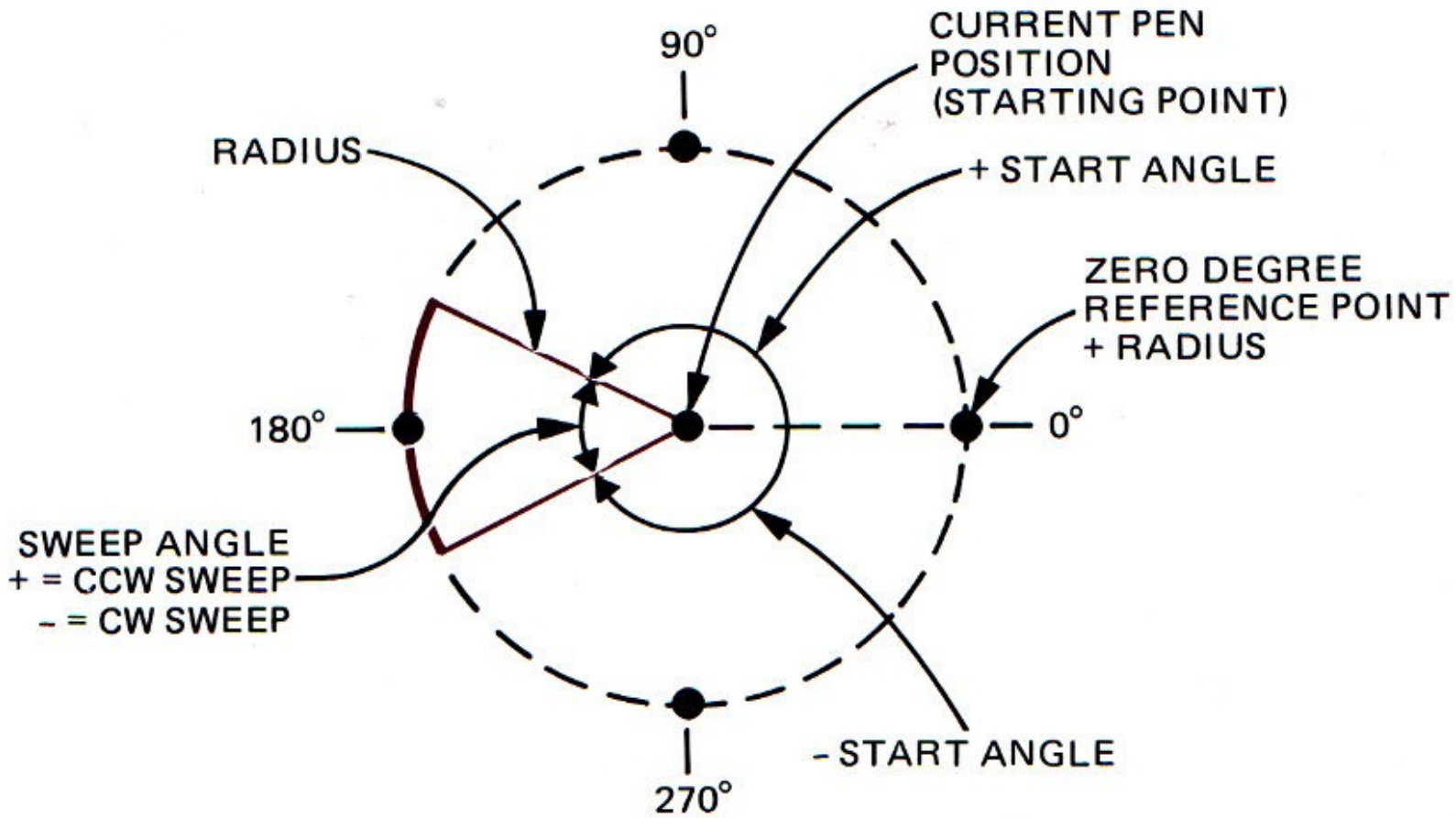
DESCRIPTION The shade wedge instruction, WG, is used to define and shade any arc segment of a circle of a specified radius.

USES This instruction is used with the FT and PT instructions to produce individual arc wedges that can be combined to create a pie chart. It is also possible to draw triangles, diamonds, pentagons, hexagons, and octagons with this instruciton.

SYNTAX *WG* radius, start angle, sweep angle (,chord angle)
terminator

https://archive.org/details/HP7475AOperationManual

~~Human~~ *Engineer* Writable



https://archive.org/details/HP7475AOperationManual

Programming Languages: 1970s

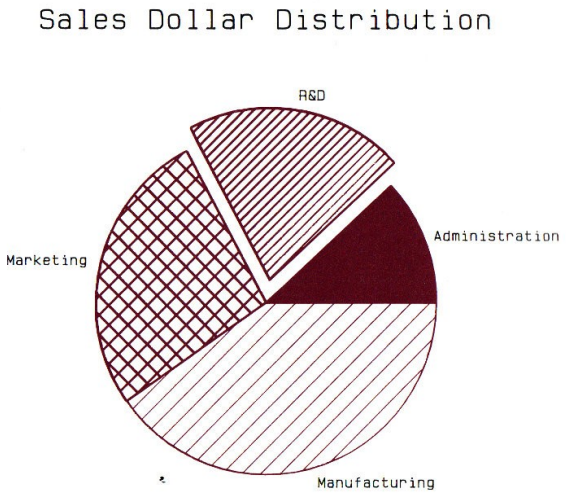
1976	Mesa	Xerox PARC	ALGOL
1976	SAM76	Claude A.R. Kagan	LISP, TRAC
1976	Ratfor	Brian Kernighan	C, FORTRAN
1976	S	John Chambers at Bell Labs	APL, PPL, Scheme
1976	SAS	SAS Institute	
1976	Integer BASIC	Steve Wozniak	BASIC
1977	FP	John Backus	none (unique language)
1977	Bourne Shell (*sh*)	Stephen R. Bourne	none (unique language)
1977	Commodore BASIC	Jack Tramiel	BASIC
1977	IDL	David Stern of Research Systems Inc	Fortran
1977	Standard MUMPS		MUMPS
1977	Icon (concept)	Ralph Griswold	SNOBOL
1977	Red	Benjamin M. Brosgol *et al.* at Intermetrics for US Dept of Defense	ALGOL 68, CS-4
1977	Blue	John B. Goodenough[7][8] *et al.* at SofTech for US Dept of Defense	ALGOL 68
1977	Yellow	Jay Spitzen *et al.* at SRI International for US Dept of Defense	ALGOL 68
1977	Euclid	Butler Lampson at Xerox Parc, Ric Holt and James Cordy at University of Toronto	
1977	Applesoft BASIC	Marc McDonald and Ric Weiland	BASIC
1978	RAPT	Pat Ambler and Robin Popplestone	APT
1978	C shell	Bill Joy	C
1978	RPG III	IBM	FARGO, RPG, RPG II
1978	HAL/S	designed by Intermetrics for NASA	XPL
1978	Applesoft II BASIC	Marc McDonald and Ric Weiland	Applesoft BASIC
1975	Irvine Dataflow (implementation)	Arvind and Gostelow, University of California, Irvine	

https://en.wikipedia.org/wiki/Timeline_of_programming_languages

Engineer Writable

Completion of Pie Chart

At the completion of the program, the scaling points are reset to their default location, the pen is raised and put away, and the finished plot is presented for viewing.

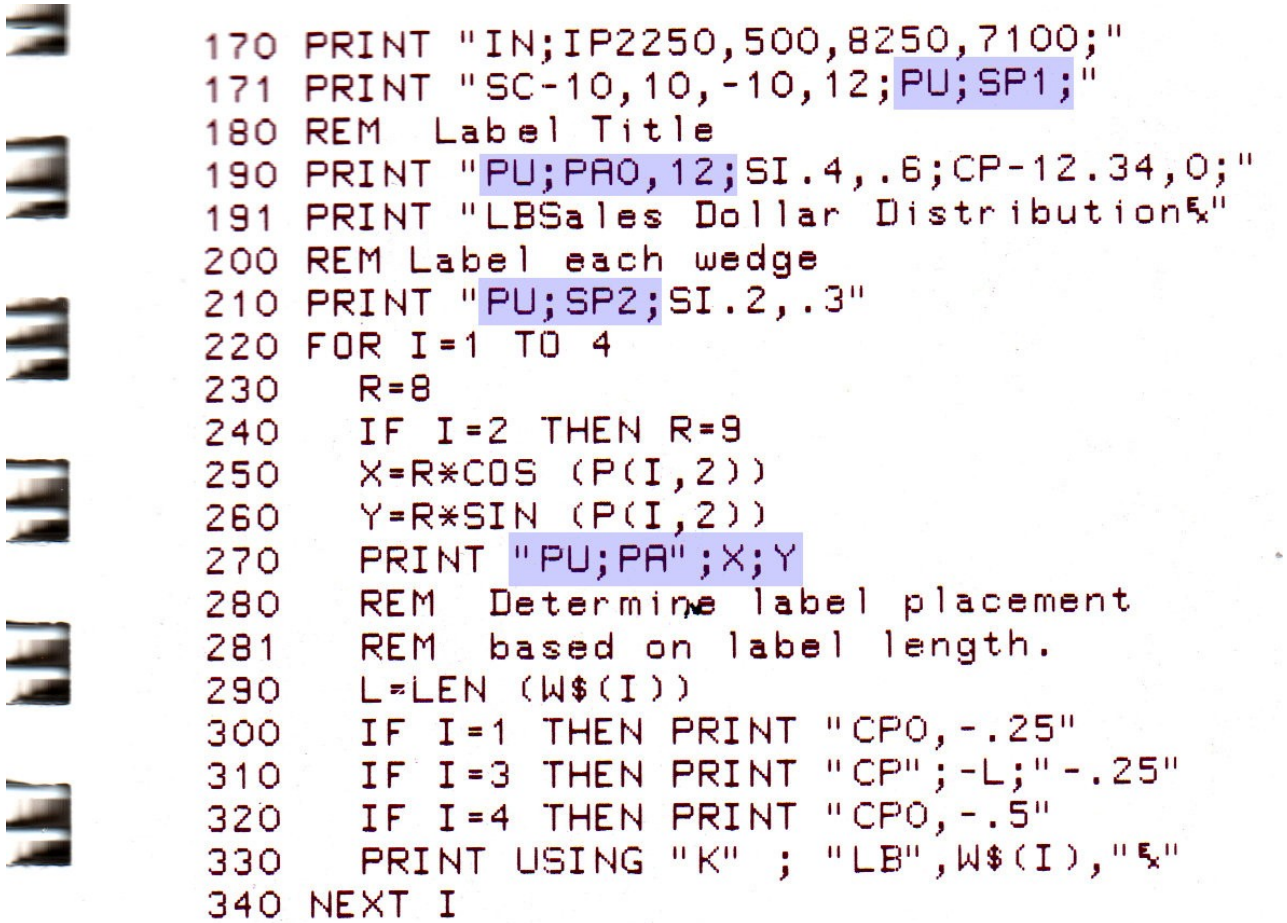



```
10 PRINTER IS 705
20 REM  PIE CHART EXAMPLE
30 REM
40 OPTION BASE 1
50 DIM W$(4)[30],P(4,3)
60 DEG
70 REM
80 DATA 0,21.5,43,43,80.5,118,118,166.5
81 DATA 215,215,287,360
90 DATA Administration,R&D,Marketing,Manufacturing
95 REM  Read start, mid-point, and stop angle
96 REM  for each segment.
110 FOR I=1 TO 4
120 FOR J=1 TO 3
130 READ P(I,J)
140 NEXT J
150 NEXT I
155    READ W$(1),W$(2),W$(3),W$(4)
160 REM
```

```
170 PRINT "IN;IP2250,500,8250,7100;"
171 PRINT "SC-10,10,-10,12;PU;SP1;"
180 REM  Label Title
190 PRINT "PU;PA0,12;SI.4,.6;CP-12.34,0;"
191 PRINT "LBSales Dollar Distribution"&CHR$(3)
200 REM Label each wedge
210 PRINT "PU;SP2;SI.2,.3"
220 FOR I=1 TO 4
230   R=8
240   IF I=2 THEN R=9
250   X=R*COS (P(I,2))
260   Y=R*SIN (P(I,2))
270   PRINT "PU;PA";X;Y
280   REM  Determine label placement
281   REM  based on label length.
290   L=LEN (W$(I))
300   IF I=1 THEN PRINT "CP0,-.25"
310   IF I=3 THEN PRINT "CP";-L;"-.25"
320   IF I=4 THEN PRINT "CP0,-.5"
330   PRINT USING "K" ; "LB",W$(I),CHR$(3)
340 NEXT I
350 REM  Draw and fill the wedges
351 REM  using pens 3,4,5 and 6.
360 FOR I=1 TO 4
370   PRINT "SP";I+2;"PT.7"
380   X=0
390   Y=0
400   IF I=2 THEN X=COS (P(I,2))
410   IF I=2 THEN Y=SIN (P(I,2))
420   IF I=1 THEN PRINT "FT1"
430   IF I=2 THEN PRINT "FT3,.3,45"
440   IF I=3 THEN PRINT "FT4,.6,45"
450   IF I=4 THEN PRINT "FT3,.6,45"
451 REM Compute the sweep angle.
460   S=P(I,3)-P(I,1)
461 REM Fill the wedge.
470   PRINT "PU;PA";X;Y;"WG7.5";P(I,1);S
471 REM Edge the wedge.
480   PRINT "PU;PA";X;Y;"EW7.5";P(I,1);S
490 NEXT I
500 REM Finish
510 PRINT "IP;SC0,1,0,1;PU;PA0,1;SP0"
520 END
```

https://archive.org/details/HP7475AOperationManual

Engineer Writable



```
170 PRINT "IN;IP2250,500,8250,7100;"
171 PRINT "SC-10,10,-10,12;PU;SP1;"
180 REM  Label Title
190 PRINT "PU;PA0,12;SI.4,.6;CP-12.34,0;"
191 PRINT "LBSales Dollar Distribution"
200 REM Label each wedge
210 PRINT "PU;SP2;SI.2,.3"
220 FOR I=1 TO 4
230   R=8
240   IF I=2 THEN R=9
250   X=R*COS (P(I,2))
260   Y=R*SIN (P(I,2))
270   PRINT "PU;PA";X;Y
280   REM  Determine label placement
281   REM  based on label length.
290   L=LEN (W$(I))
300   IF I=1 THEN PRINT "CP0,-.25"
310   IF I=3 THEN PRINT "CP";-L;"-.25"
320   IF I=4 THEN PRINT "CP0,-.5"
330   PRINT USING "K" ; "LB",W$(I),""
340 NEXT I
```

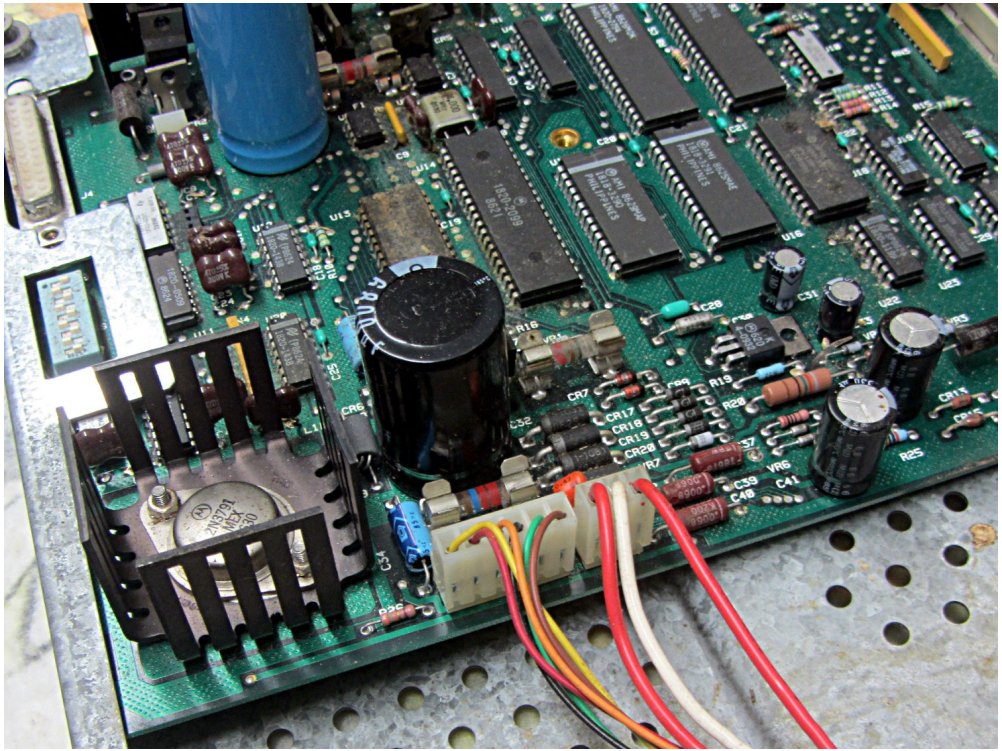
https://archive.org/details/HP7475AOperationManual

Chiplotle: Python HPGL API

“A way to control

your grungy old pen plotters

with your shiny new laptop!”



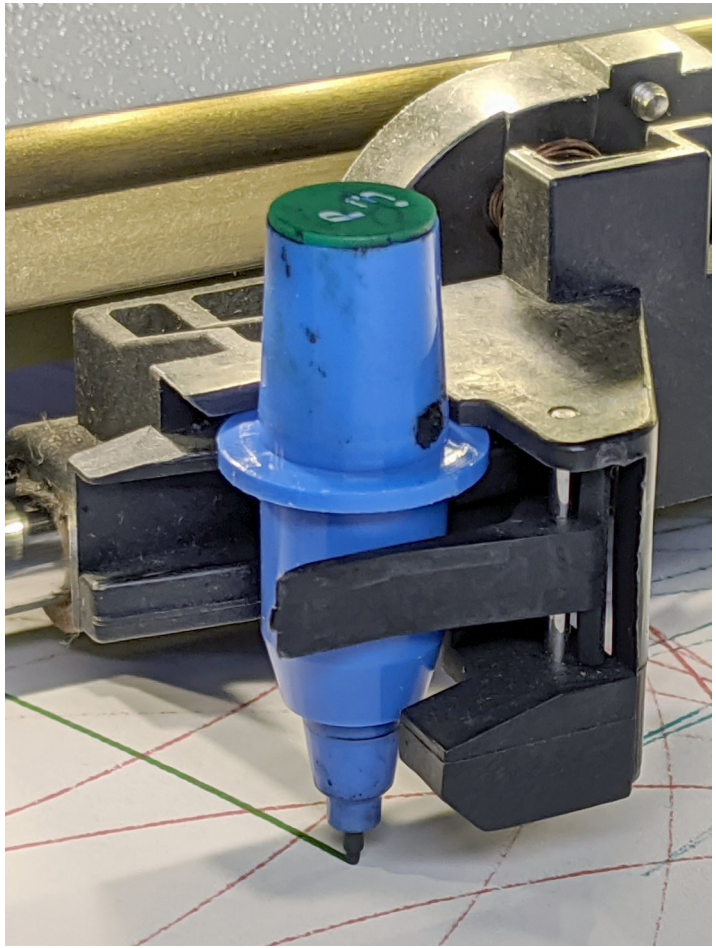
http://sites.music.columbia.edu/cmc/chiplotle/

Python / Chiplotle vs. HPGL

`plt = instantiate_plotters()[0]`	ESC.B ESC.(; ← The First Smiley?
	IN;
	OW;OW;OW;OW;
`plt.write(chr(27) + '.H200:')`	ESC.H200:;
	SC;
	OW;OW;OW;OW;
`plt.set_origin_center()`	IP0,0,16640,10365;
	OW;OW;
	SC-8320,8320,-5182,5182;
`plt.write(hpgl.SI(tscale * 0.285, tscale * 0.375))`	SI0.13,0.17;
`plt.write(hpgl.VS(8))`	VS8;
`plt.write(hpgl.PA([(legendx, legendy)]))`	PA5320,-4282;
`plt.select_pen(pen)`	SP1;
`plt.write(hpgl.PA([(legendx, legendy)]))`	PA5320,-4282;
`plt.write(hpgl.LB("Started " + str(datetime.today())))`	LBStarted 2020-01-09 18:03:57.494617ETX;
`plt.select_pen(pen)`	SP1;
`plt.write(hpgl.PA([(legendx, legendy - 100 * pen)]))`	PA5320,-4382;
`plt.write(hpgl.LB("Pen {0}: ".format(pen)))`	LBPen 1: ETX;
`plt.select_pen(1)`	SP1;
`plt.write(hpgl.LB("m={0:.1f} n1={1:.2f} n2=n3={2:.2f}".format(m, n1, n2)))`	LBm=1.9 n1=0.71 n2=n3=0.26ETX;
`plt.select_pen(pen)`	SP1;
`plt.write(e)`	PU;
	PA8320.00,0.00;
	PD;
	PA8320.00,0.00,
	6283.71,24.59,
	<<< snippage >>>

HP *Graphics* Language

- Detailed hardware control
- Reasonable graphic primitives
- Not Turing complete
  - No variables / memory
  - No arithmetic / logic
  - No iteration
  - No flow control
- *Nobody* writes HPGL today!



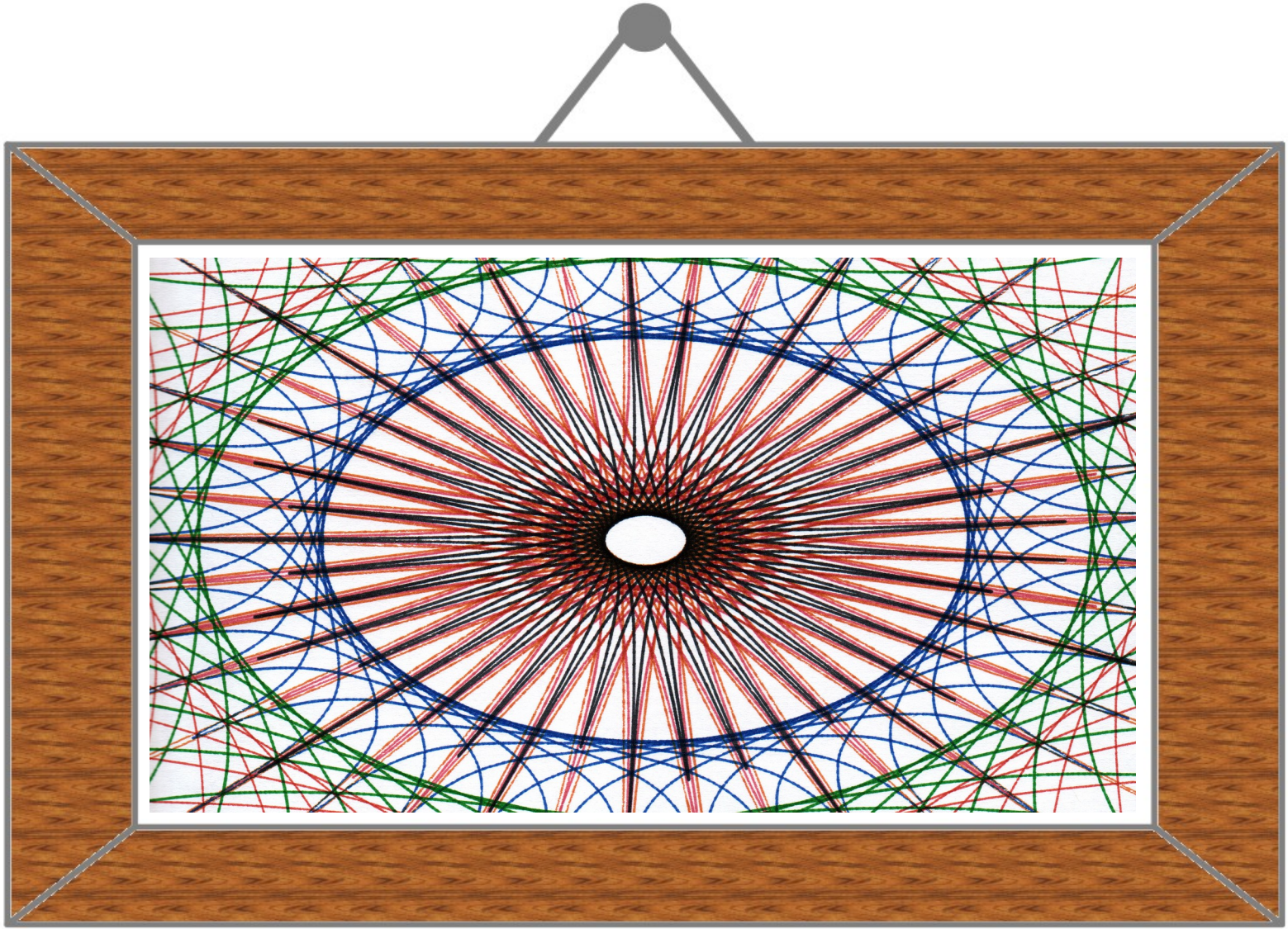
https://en.wikipedia.org/wiki/HP-GL

HP *Graphics* Language

- Still in use (!)
  - Craft cutting machines
  - Laser cutters
  - Wide-format vinyl cutters

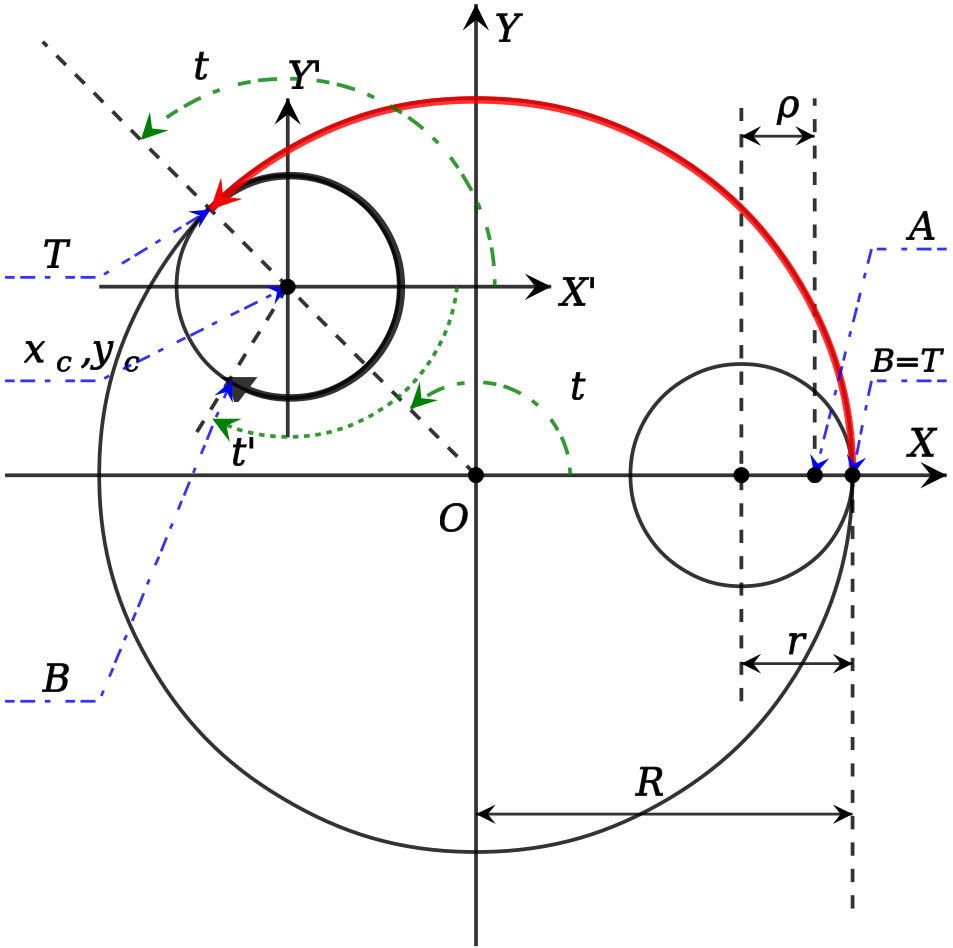
- As an output language
  - Any input language you prefer
  - Graphics direct to HPGL
  - With USB / serial port driver …

https://en.wikipedia.org/wiki/HP-GL



http://softsolder.com/2015/11/02/hp-7475a-plotter-one-button-demo-madness/

Spirograph Analysis (!)



https://en.wikipedia.org/wiki/Spirograph

Parametric Spirograph Equations

$$x(t) = R\left[(1-k)\cos t + lk\cos\frac{1-k}{k}t\right]$$

$$y(t) = R\left[(1-k)\sin t - lk\sin\frac{1-k}{k}t\right]$$

https://en.wikipedia.org/wiki/Spirograph

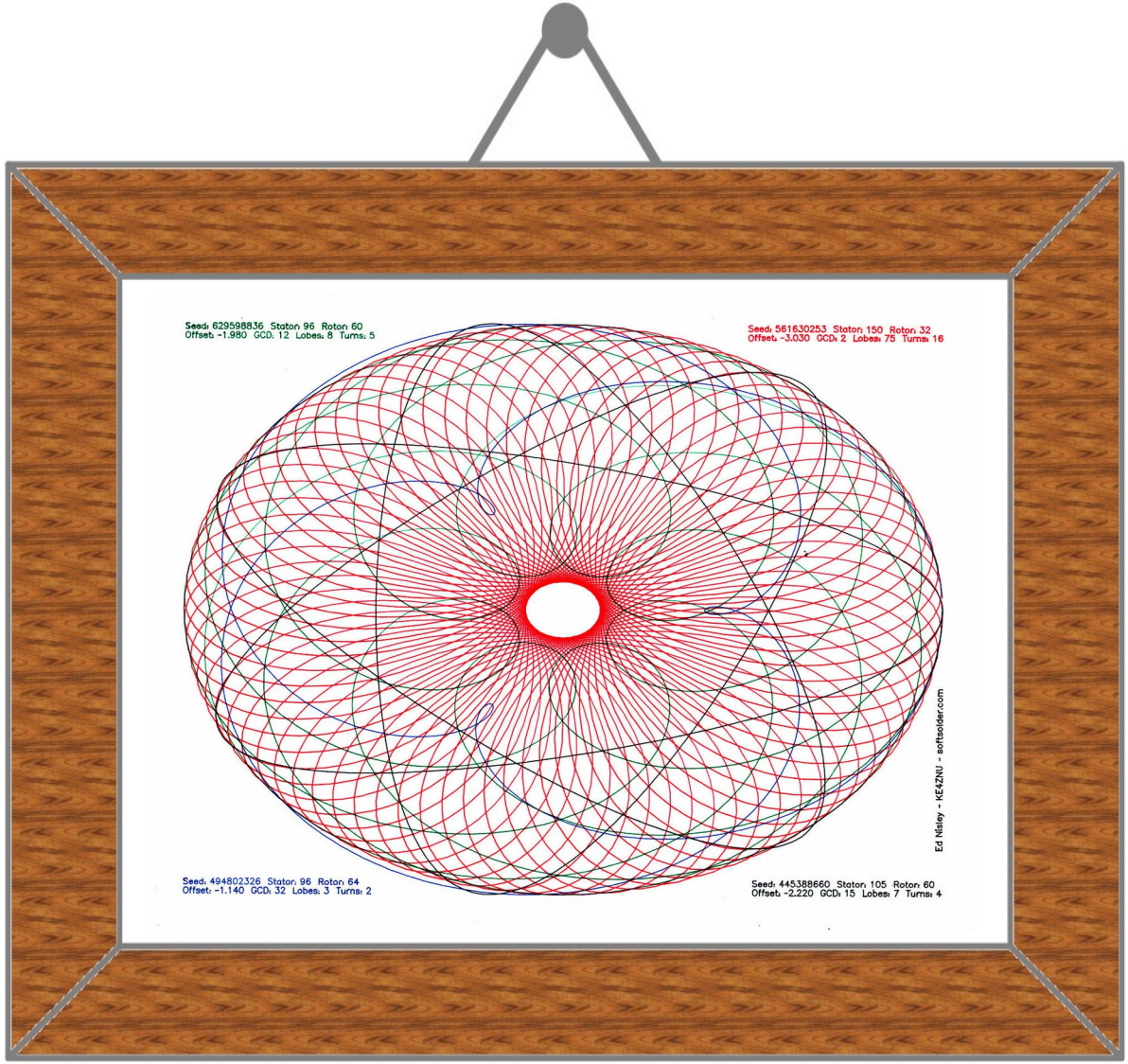
Spirograph Algorithm

```
Path = {};
Xmax = 0.0;
Xmin = 0.0;
Ymax = 0.0;
Ymin = 0.0;

for (a = 0.0deg ; a <= Turns*360deg ; a += AngleStep) {
  x = (1 - K)*cos(a) + L*K*cos(a*(1 - K)/K);
  if   (x > Xmax) {Xmax = x;}
  elif (x < Xmin) {Xmin = x;}

  y = (1 - K)*sin(a) - L*K*sin(a*(1 - K)/K);
  if   (y > Ymax) {Ymax = y;}
  elif (y < Ymin) {Ymin = y;}

  Path += {[x,y]};
}
```

http://softsolder.com/2017/12/26/mpcnc-spirograph-generator-with-tool-changes/

Math ➔ Paper

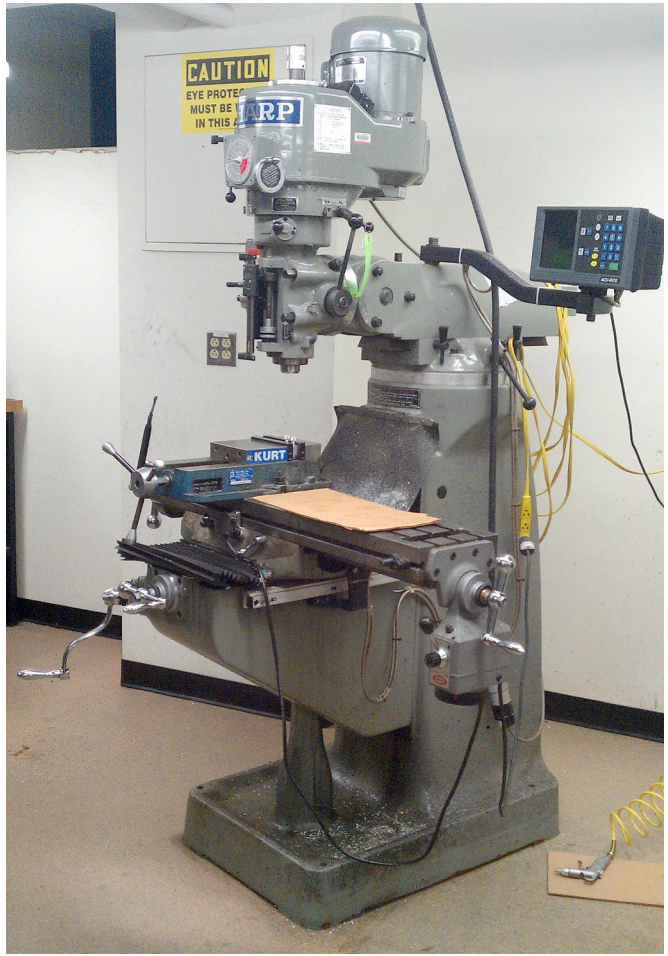
```
Xmax = max(Xmax,-Xmin);
Ymax = max(Ymax,-Ymin);

PlotScale = [PlotSize.x / (2*Xmax),
             PlotSize.y / (2*Ymax)];

Points = scale(Path,PlotScale);

feedrate(2400.0mm);

PlotZ = -1.00mm;

tracepath(Points, PlotZ);
```



https://softsolder.com/2020/01/02/cnc-3018xl-rotating-the-axes/

CNC Machine

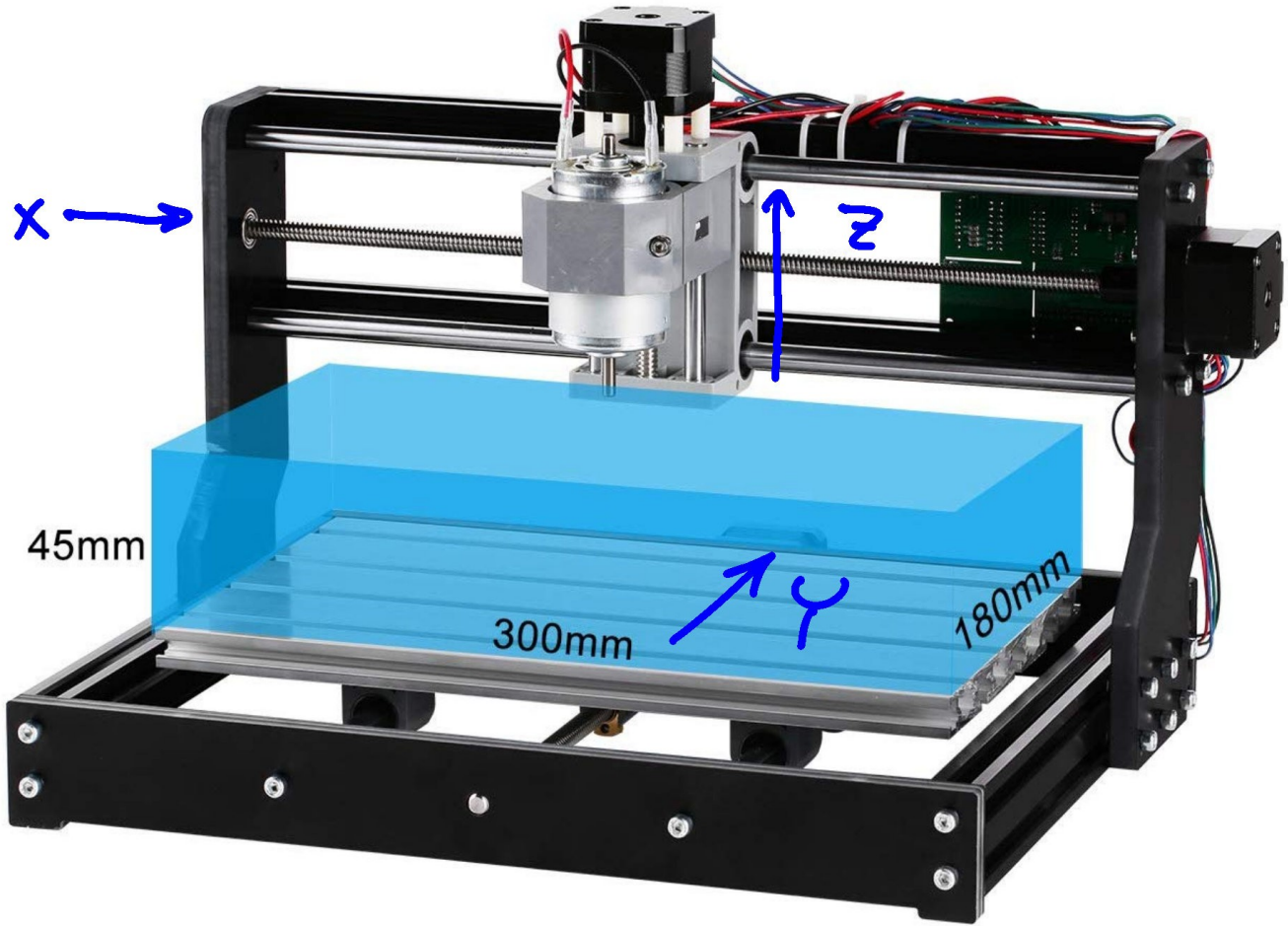
- Computer
  - Mostly after mid-1960s
- Numeric
  - Coordinates *punched in paper tape*
  - Bresenham's Algorithm: 1962 (!)
- Control
  - Hands-off = faster & repeatable
- Machine
  - Mill, lathe, drill, laser cutter …
  - 3D Printers (!)



https://commons.wikimedia.org/wiki/File:Sharp_3_Axis_Vertical_Mill_Full_View.jpg

https://en.wikipedia.org/wiki/Numerical_control

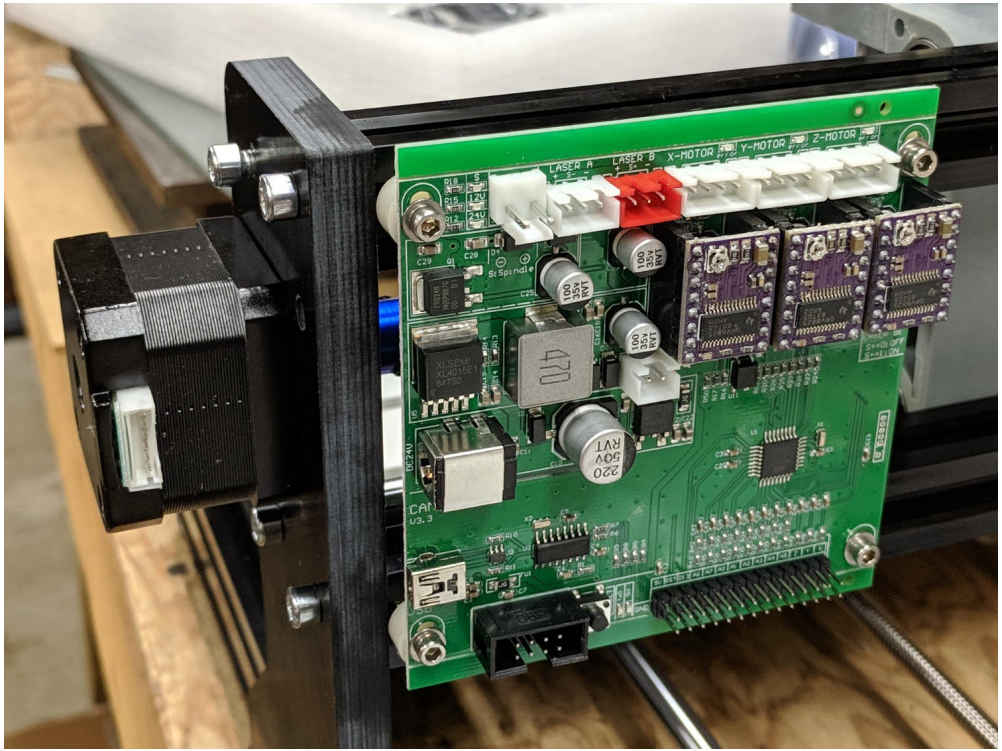
CNC 3018-Pro



https://www.amazon.com/Genmitsu-3018-PRO-Control-Engraving-300x180x45mm/dp/B07P6K9BL3

“Arduino” Controller

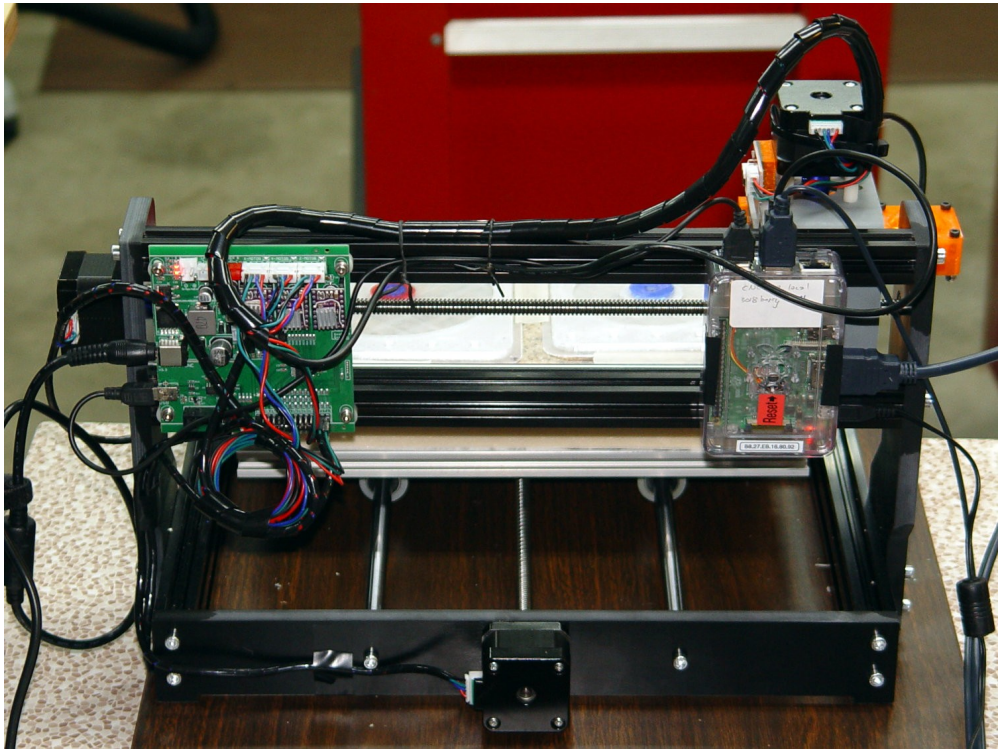
- Atmega 328 µC
  - Teleported from 1995
  - Single-chip 1980s tech
  - 32 kB program
  - 2 kB RAM
- Plug-in stepper drivers
- Header for I/O bits
- Pre-installed GRBL 1.1f
  - G-Code serial *input*
  - Motion control *output*



https://github.com/gnea/grbl

Add a Raspberry Pi

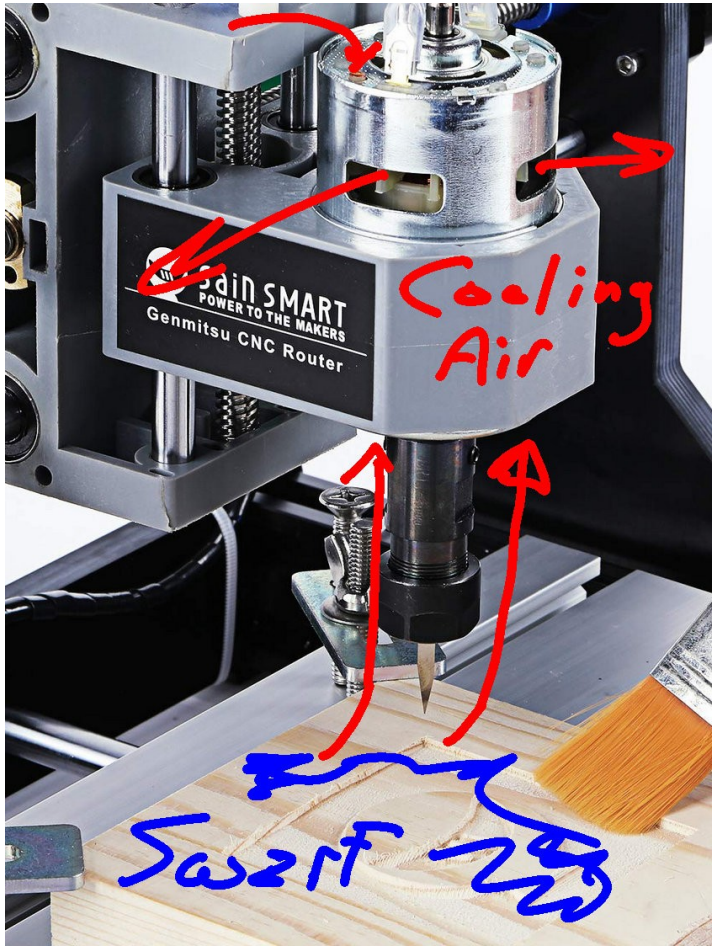
- Linux, of course
  - Raspbian + desktop GUI
  - G-Code files via network
  - VNC remote access
- bCNC User Interface
  - Manual jogging
  - Touch off XYZ=0
  - Load & run G-Code
  - *Many* additional features



https://www.raspberrypi.org/downloads/raspbian/
https://github.com/vlachoudis/bCNC

CNC 3018-Pro

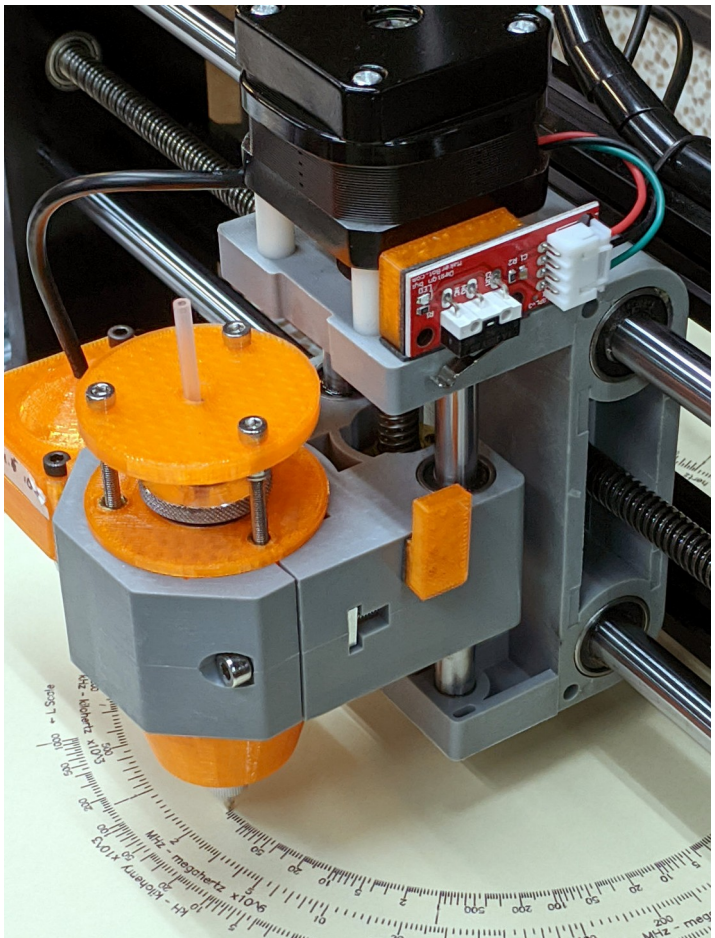
- ≅ 3D printer hardware
  - Cheap & readily available
  - Vast array of clones / knockoffs
  - Unpredictable quality
- ≅ Rigid & aligned & stable
  - 2020/2040 Al extrusion frame
  - Stepper + leadscrew drive
  - Spindle motor is … sketchy
- Think of it as a parts kit
  - For *any* CNC-like project
- $200 to $250-ish w/ extras



https://www.amazon.com/Genmitsu-3018-PRO-Control-Engraving-300x180x45mm/dp/B07P6K9BL3

CNC 3018-“XL”

- Hardware & config tweaks
  - Longer platform: “CNC 4330”
  - Moah Speed!!!
- Home switches
- Run-Halt buttons
- Alignment Probe Camera
- Custom tooling
  - Drag knife
  - Ball-point pens (cheap vs. nice)
  - Diamond point drag engraver



https://softsolder.com/2019/09/19/cnc-3018-pro-drv8825-drivers-at-the-edge-of-madness/

G-Code

The first implementation of a

numerical control programming language

was developed at the

MIT Servomechanisms Laboratory

in the late 1950s

https://en.wikipedia.org/wiki/G-code

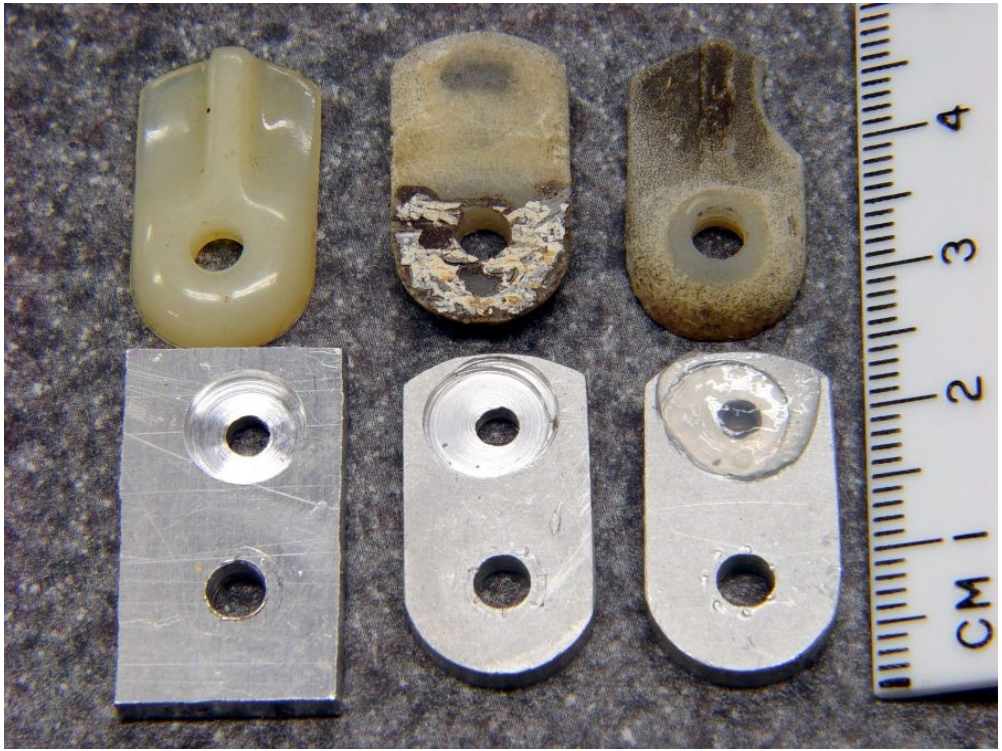
Programming Languages: 1950s

1954	IPL I (concept)	Allen Newell, Cliff Shaw, Herbert A. Simon	none (unique language)
1955	FLOW-MATIC	Team led by Grace Hopper at UNIVAC	A-0
1955	BACAIC	M. Grems and R. Porter	
1955	PACT I	SHARE	FORTRAN, A-2
1955	Freiburger Code[4][5]	University of Freiburg	N/A
1955–56	Sequentielle Formelübersetzung	Fritz Bauer and Karl Samelson	Boehm
1955–56	IT	Team led by Alan Perlis	Laning and Zierler
1955	PRINT	IBM	
1958	IPL II (implementation)	Allen Newell, Cliff Shaw, Herbert A. Simon	IPL I
1956–58	LISP (concept)	John McCarthy	IPL
1957	COMTRAN	Bob Bemer	FLOW-MATIC
1957	GEORGE	Charles Leonard Hamblin	none (unique language)
1957	Fortran I (implementation)	John W. Backus at IBM	FORTRAN
1957–58	UNICODE	Remington Rand UNIVAC	MATH-MATIC
1957	COMIT (concept)	Victor Yngve	none (unique language)
1958	Fortran II	Team led by John W. Backus at IBM	FORTRAN I
1958	ALGOL 58 (IAL)	ACM/GAMM	FORTRAN, IT, Sequentielle Formelübersetzung
1958	IPL V	Allen Newell, Cliff Shaw, Herbert A. Simon	IPL II
1959	APT	Douglas T. Ross	
1959	FACT	Fletcher R. Jones, Roy Nutt, Robert L. Patrick	none (unique language)
1959	COBOL (concept)	The CODASYL Committee	FLOW-MATIC, COMTRAN, FACT
1959	JOVIAL	Jules Schwartz at SDC	ALGOL 58
1959	LISP (implementation)	John McCarthy	IPL
1959	MAD – Michigan Algorithm Decoder	Bruce Arden, Bernard Galler, and Robert M. Graham	ALGOL 58
1959	TRAC (concept)	Calvin Mooers	

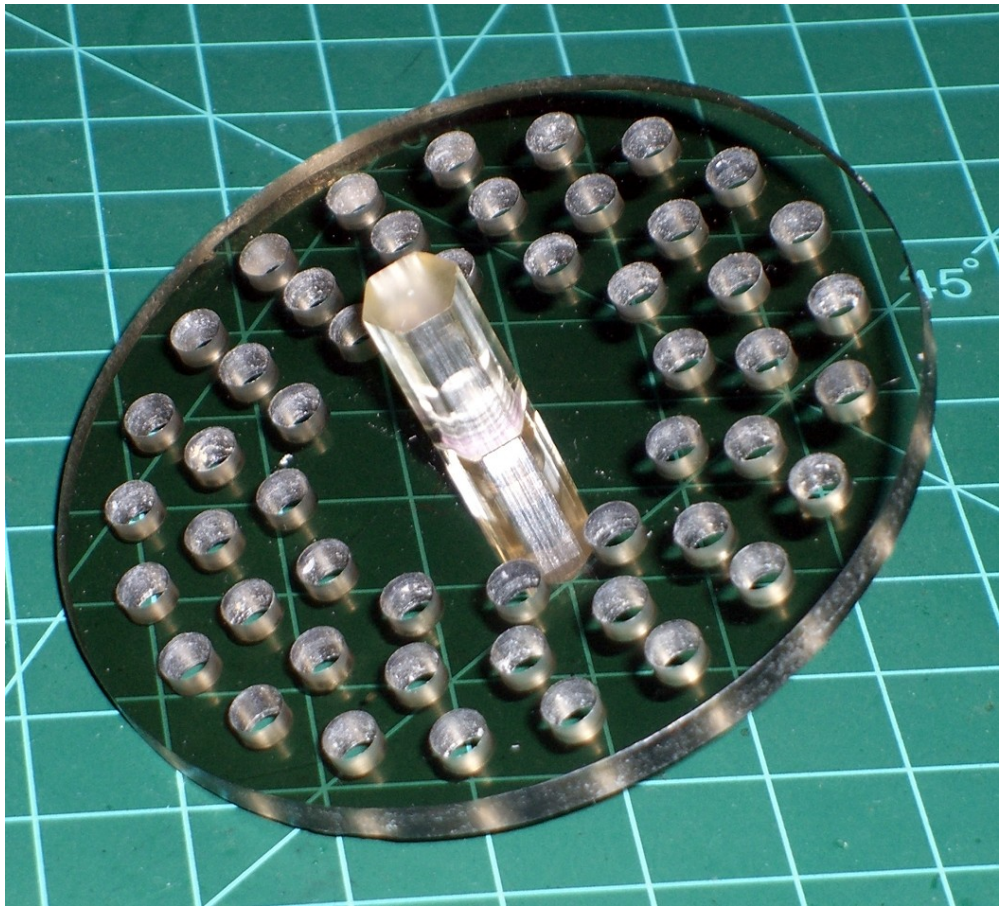
https://en.wikipedia.org/wiki/Timeline_of_programming_languages

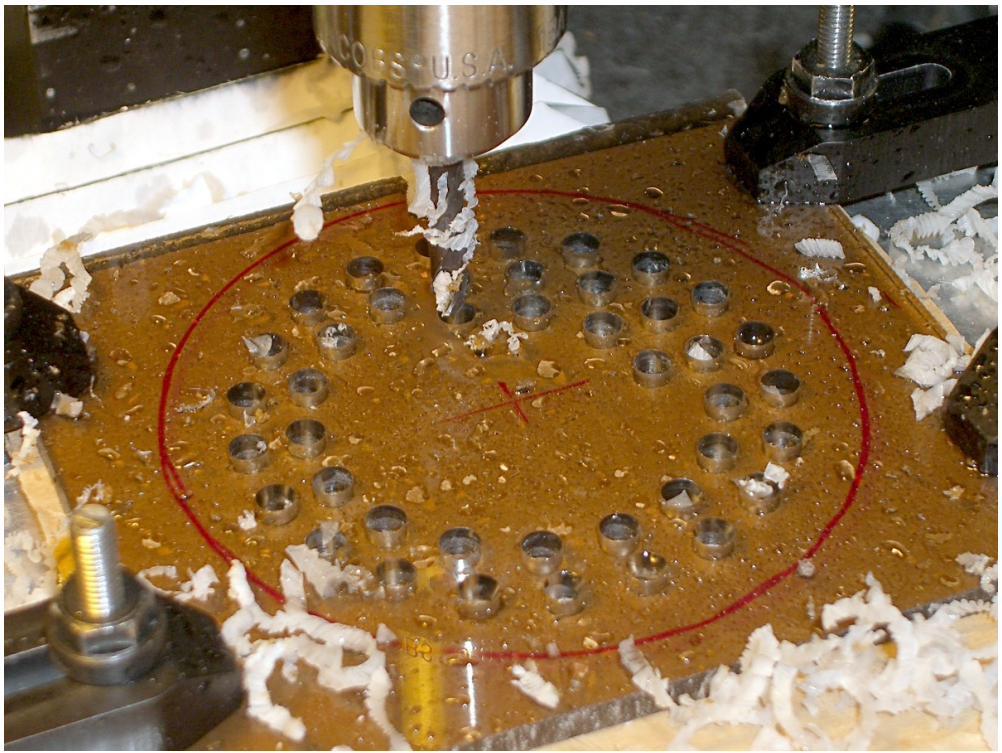
Window Latch

```
#1011 =  9.0 (traverse clearance)
#1015 =  3.2 (piece thickness)

#1024 = 13.8 (X piece side length)
#1025 = 12.6 (Y piece width)

#1026 =  2.3 (pad endcap arc height - m dimension)

#1027 =  [#1025 / 2] (radius of mtg endcap - half of #1025)

#1028 =  [ [#1026 ** 2 + [[#1025 ** 2] / 4]] / [2 * #1026] ]
                        (radius of pad endcap)

#1029 = [#1028 - #1026]  (X offset from end of straight side
                         to ctr of pad arc)

G21      (metric units)
F100     (milling feed - mm/min)
S2500    (spindle speed - rev/min)

G0 X0 Y0 Z[#1011]            (to origin at traverse level)
G0 X0 Y[0 - #1027] Z0    (move to start)

G1 Z[0 - #1015]          (down to cut level)

G1 X[#1024]              (cut front edge)
G3 Y[#1027] I[0 - #1029] J[#1027] (cut pad endcap)
G1 X0                    (cut far side)
G3 Y[0 - #1027] I0 J[0 - #1027]   (cut mounting endcap)
G0 Z[#1011]              (up to surface)
M2                       (stop)
```



CNC Sink Strainer





https://softsolder.com/2013/06/07/search-engine-optimization-which-one-is-not-like-the-other-ones/

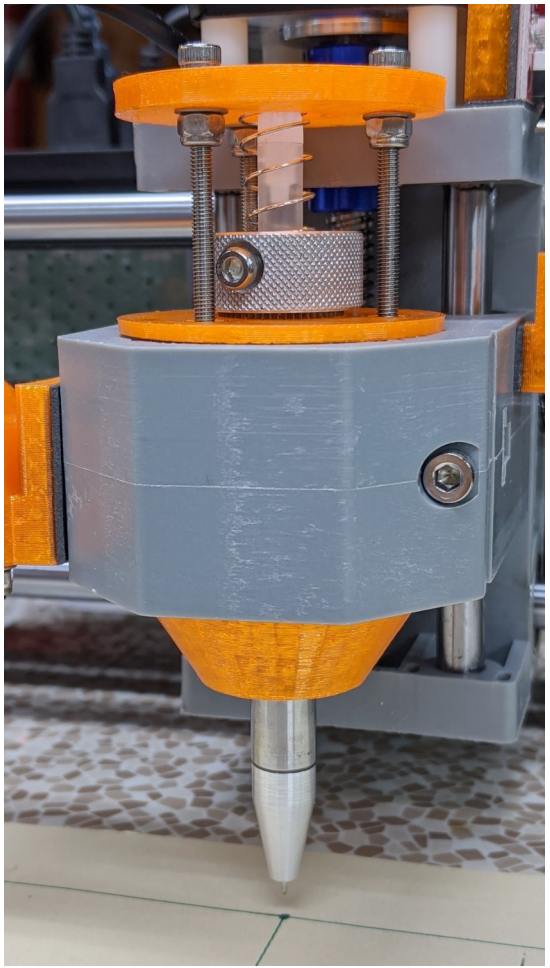
Drill Holes Around a Circle

```
( Parameters: #1=Circle radius, #2=Hole dia, #3=Phase )

O9000 SUB

#<HoleWidthAngle> = [2 * ASIN [#2/[2 * #1]]]        (angle subtended by hole dia)
#<NumHoles>  = FIX[360 / [2 * #<HoleWidthAngle>]]   (integer number of holes + spaces)
#<HoleAngle> = [360 / #<NumHoles>]                  (angle between successive holes)

#<DrillAngle> = [#3 * #<HoleAngle> / 2]             (starting angle for drilling)

O9010 DO

#<HoleX> = [#1 * COS[#<DrillAngle>]]                (find hole coordinates)
#<HoleY> = [#1 * SIN[#<DrillAngle>]]
G81 X#<HoleX> Y#<HoleY> Z[0 - #<_DrillDepth>] R#<_TraverseZ> F#<_DrillFeed>

#<DrillAngle> = [#<DrillAngle> + #<HoleAngle>]      (step to next hole)
O9010 WHILE [360 GT #<DrillAngle>]

O9000 ENDSUB
```

GCMC vs. G-Code

```
comment("Tool change: ",Pen);
toolchange(Pen);
feedrate(2400.0mm);
tracepath(Points, PlotZ);
feedrate(2400mm);
comment("Legend begins");
engrave(placepath,TravelZ,PlotZ);
goto([-,-,SafeZ]);
goto([0,0,-]);
```

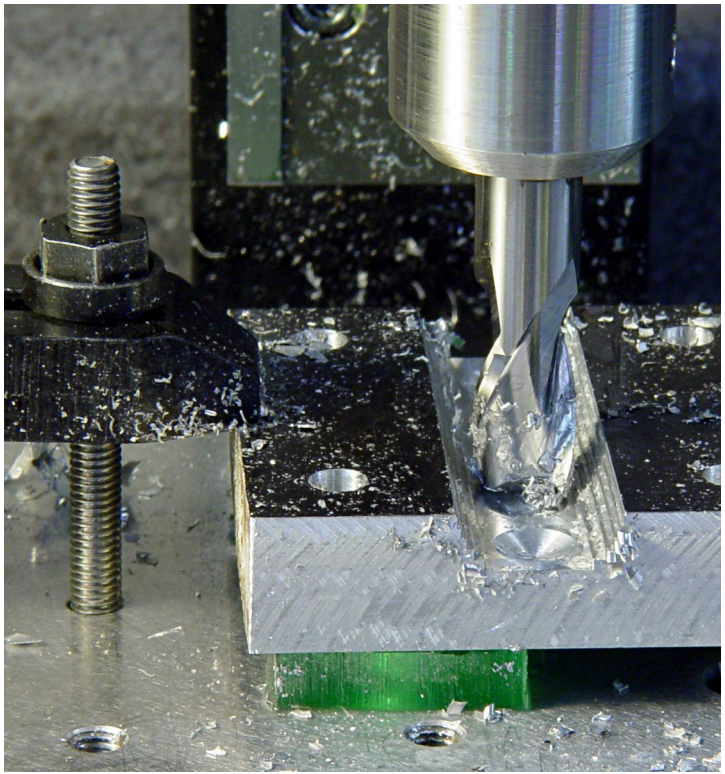
```
(Tool change: 1)
T1
M6
F2400.000
(-- tracepath at Z=-1.000mm --)
G0 X-62.859 Y0.000
G1 Z-1.000
G1 X-62.846 Y2.763
<< ≅2000 lines of snippage>>
G1 X-62.859 Y0.000
G1 Z0.000
(-- tracepath end --)
F2400.000
(Legend begins)
G0 Z1.000
G0 X63.571 Y-92.250
G0 X65.190 Y-90.536
G1 Z-1.000
G1 X65.000 Y-90.345
G1 X64.714 Y-90.250
G1 X64.333 Y-90.250
<< ≅2000 lines of snippage>>
G0 Z10.000
G0 X0.000 Y0.000
```



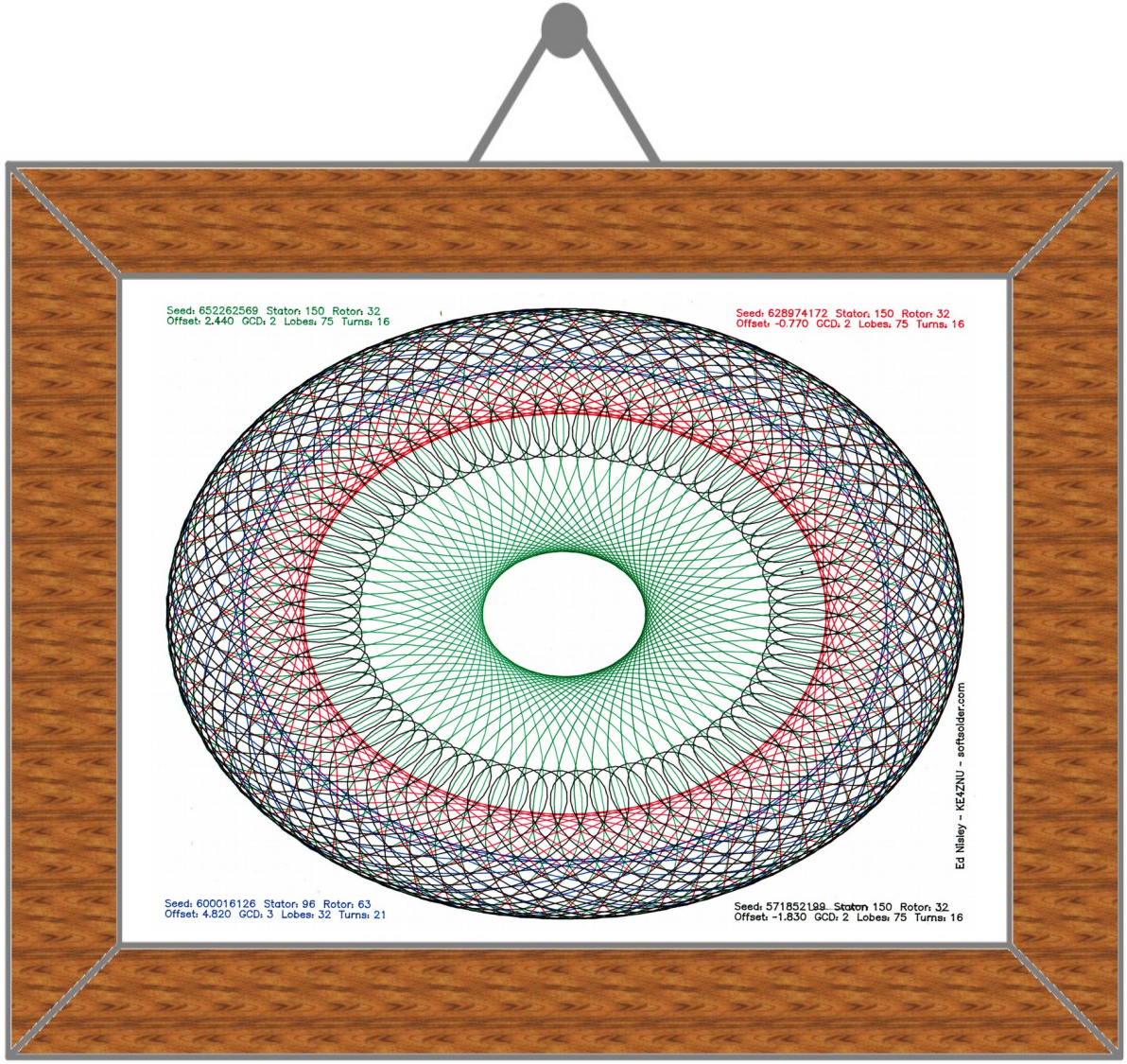
http://www.vagrearg.org/content/gcmc

G-Code Language

- Exquisite hardware control
  - You *must* do it all
- Turing complete
  - All dialects do it *differently*
  - Blinding syntactic noise
- G-Code still in use today
  - Solid model compiler output
    - Consumer-grade 3D printing
  - Humans in hard mode
    - Production optimization
    - Algorithmic Art!

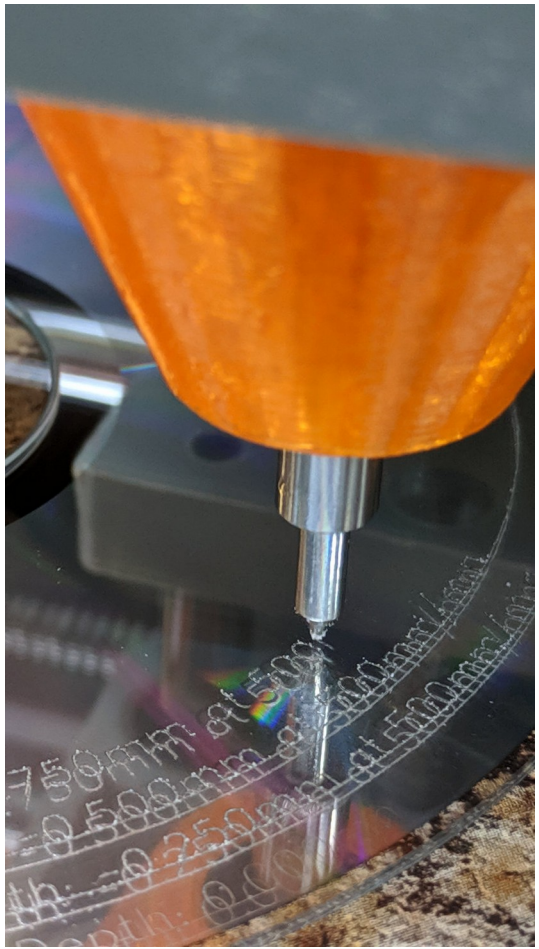


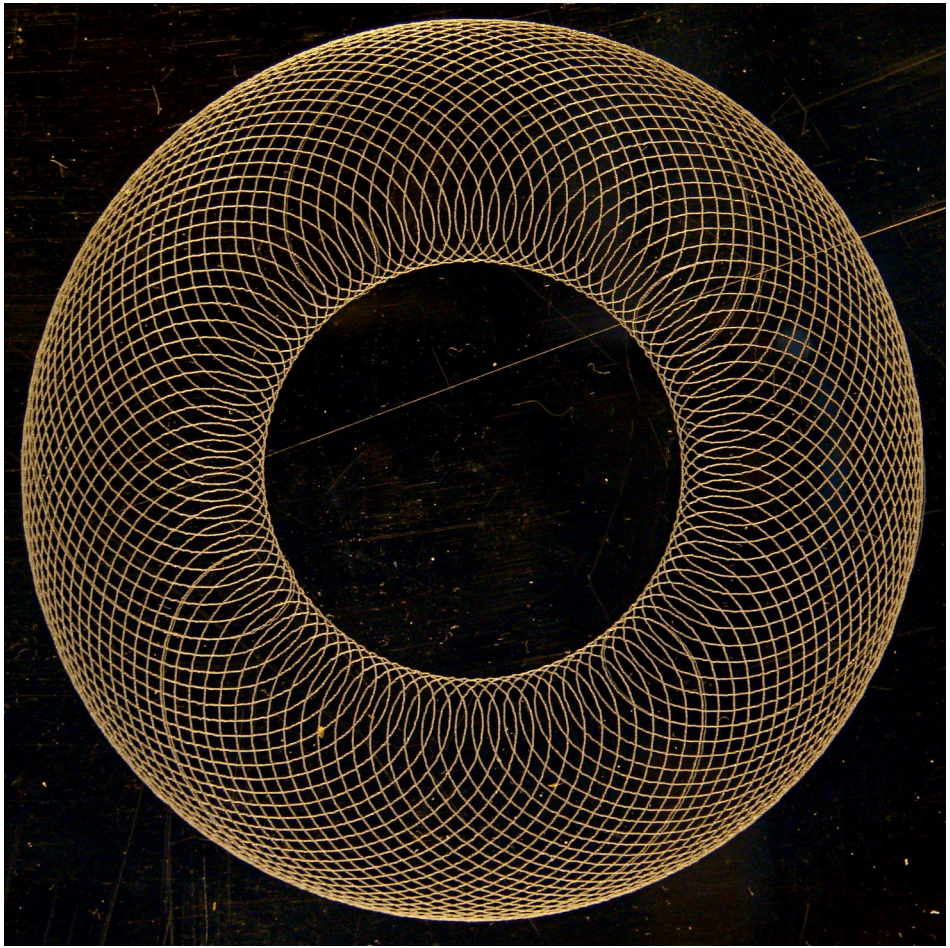
https://en.wikipedia.org/wiki/G-code



https://softsolder.com/2020/01/02/cnc-3018xl-rotating-the-axes/

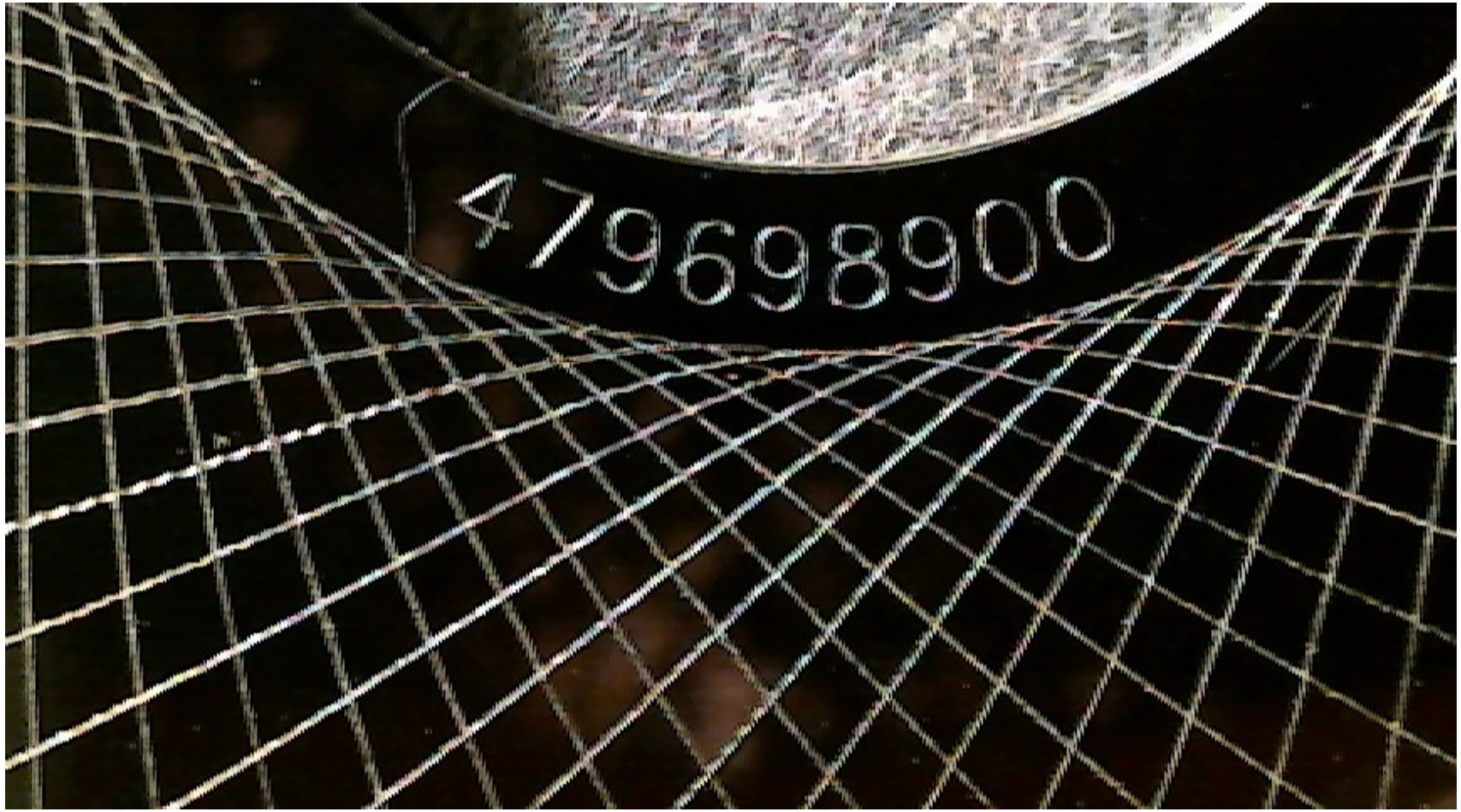
Diamond Drag Engraving



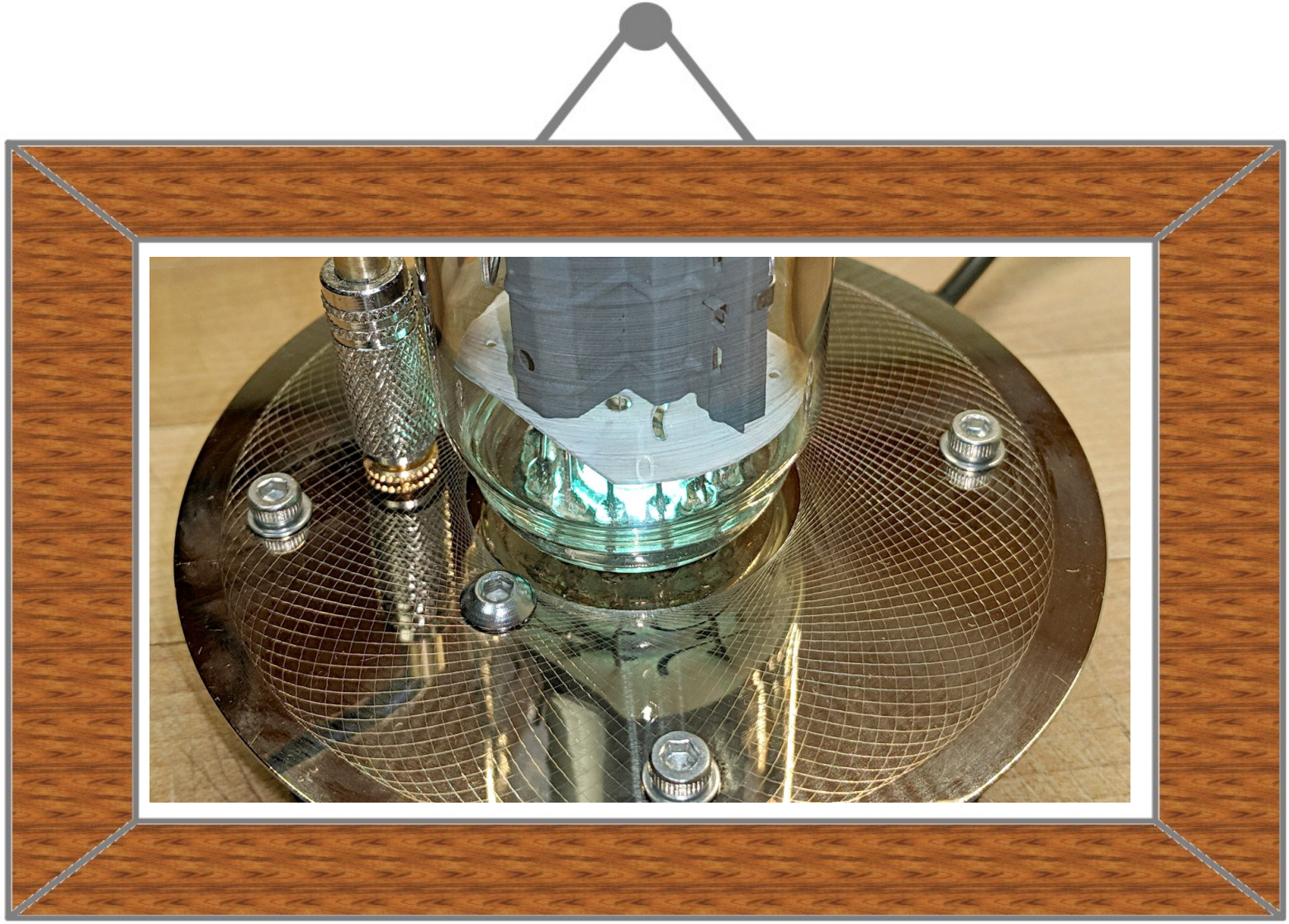


https://softsolder.com/2019/09/10/cnc-3018-pro-lm6uu-linear-bearing-diamond-drag-bit-holder/

Engraved Hard Drive Platters



https://softsolder.com/2019/12/06/diamond-drag-bit-moah-downforce/



https://softsolder.com/2018/11/23/vacuum-tube-leds-21hb5a-on-a-guilloche-platter/

Show-n-Tell
+
Q-n-A
!

https://softsolder.com/

Ed Nisley

Say “NISS-lee”, although we're on the half-essed branch of the tree

Engineer (ex PE), Hardware & Firmware Tinker, Author

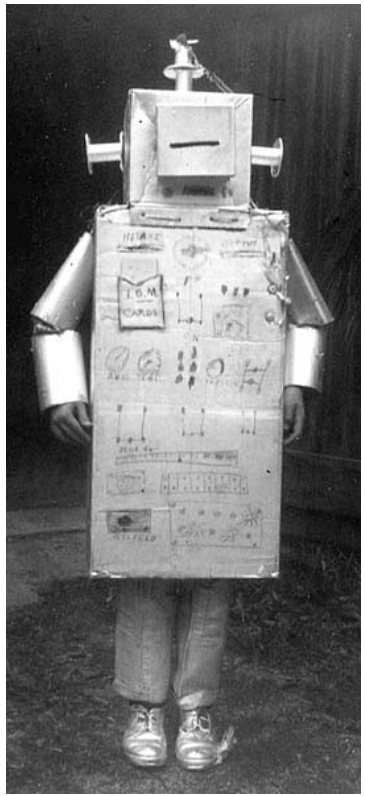
Blog The Smell of Molten Projects in the Morning - softsolder.com

Digital Machinist - www.homeshopmachinist.net
Along the G-Code Way (2008 ...) - CNC, math, 3D printing

Circuit Cellar - www.circuitcellar.com
Firmware Furnace (1988-1996) - Nasty, grubby hardware bashing
Above the Ground Plane (2001-2018) - Analog and RF stuff

Dr. Dobb’s Journal - www.ddj.com
Embedded Space (2001-2006) - All things embedded
Nisley’s Notebook (2006-2007) - Hardware & software collisions

Book! The Embedded PC's ISA Bus: Firmware, Gadgets, Practical Tricks



September
1962

If you can't read this
then
make a new friend
'way up front