In [1]:
```
from openscad3 import *
```

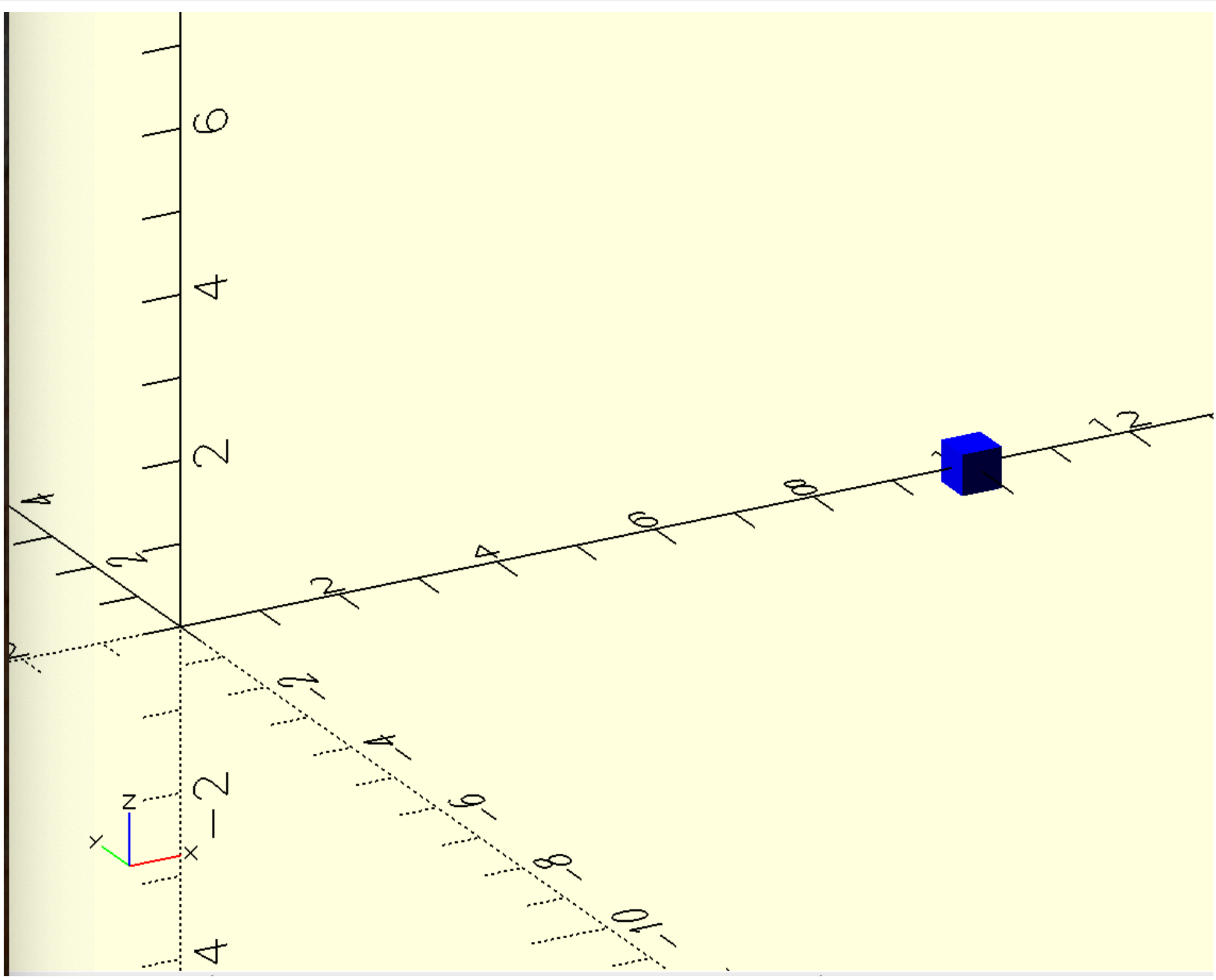
Basic of Drawing and 3D modeling with library openscad3

Basic elements are:

- point: defined by 2d or 3d coordinates
- line: defined by 2 points (2d or 3d coordinates)
- polyline: defined by more than 2 points (2d or 3d coordinates)
- surface: defined by arrangement of 2 or more lines or polylines where there is no volume
- solid: defined by arrangement of 2 or more polylines with ends closed and has volume
- plane: defined by a normal vector
- extrude along path: defined by extruding a 2d section along a 3d path
- Sculpting along path: defined by sculpting a 2d section along a 2d path
- Rotate objects: Objects can be rotated along a defined axis
- translate objects: objects can be translated by a defined vector from their relative positions
- wrapping a polyline/ surface/ solids around a path
- Intersections: between line to line, polyline to polyline/ line (2d or 3d) or between surface to surface

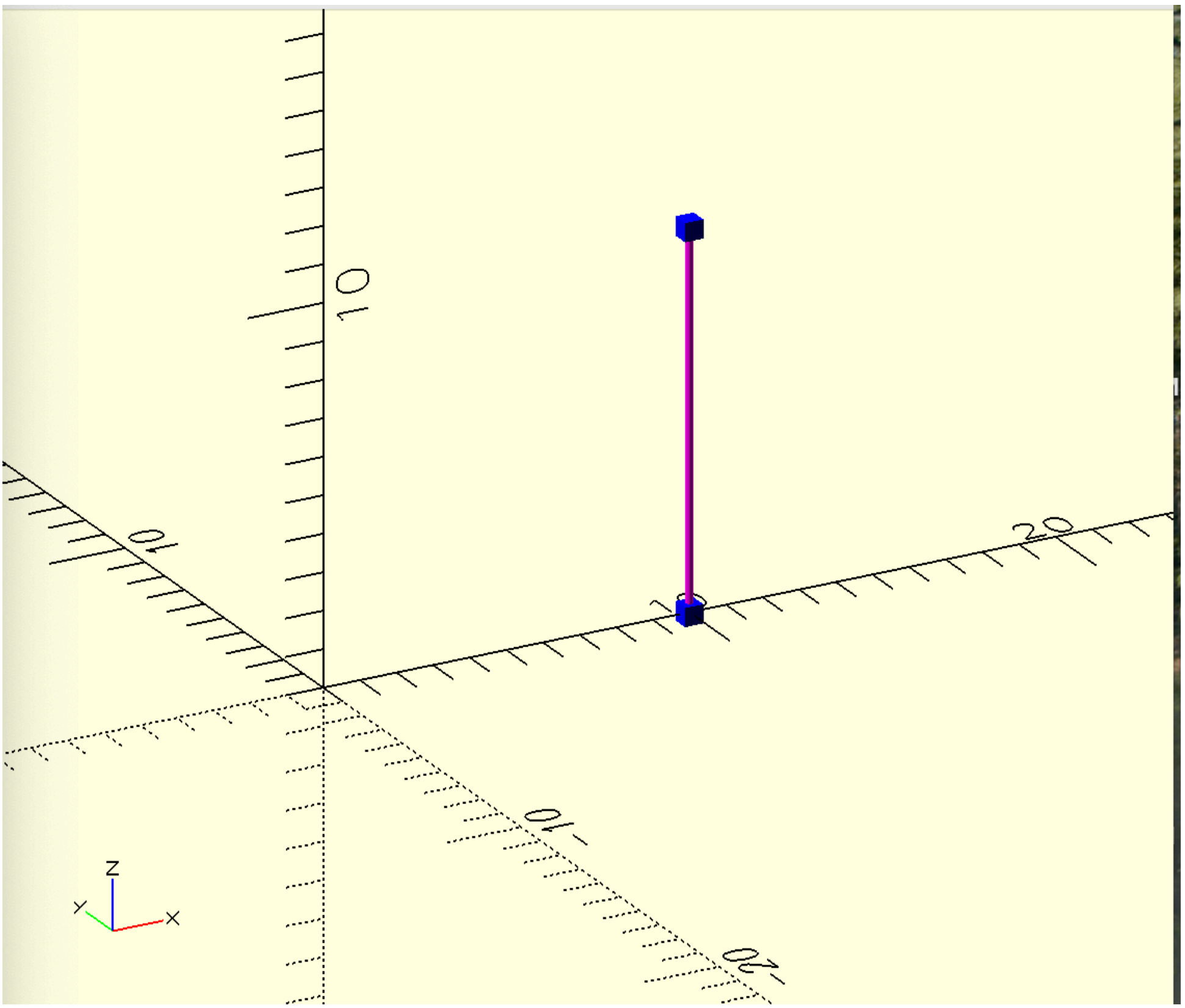
Points

In [4]:
```
p0=[10,0,0]
fileopen(f'''
// pay attention to the points module here. Points are shown as cube
// In this example cube of size 0.5 is showing the location of the p0
color("blue") points({[p0]},.5);
''')
```



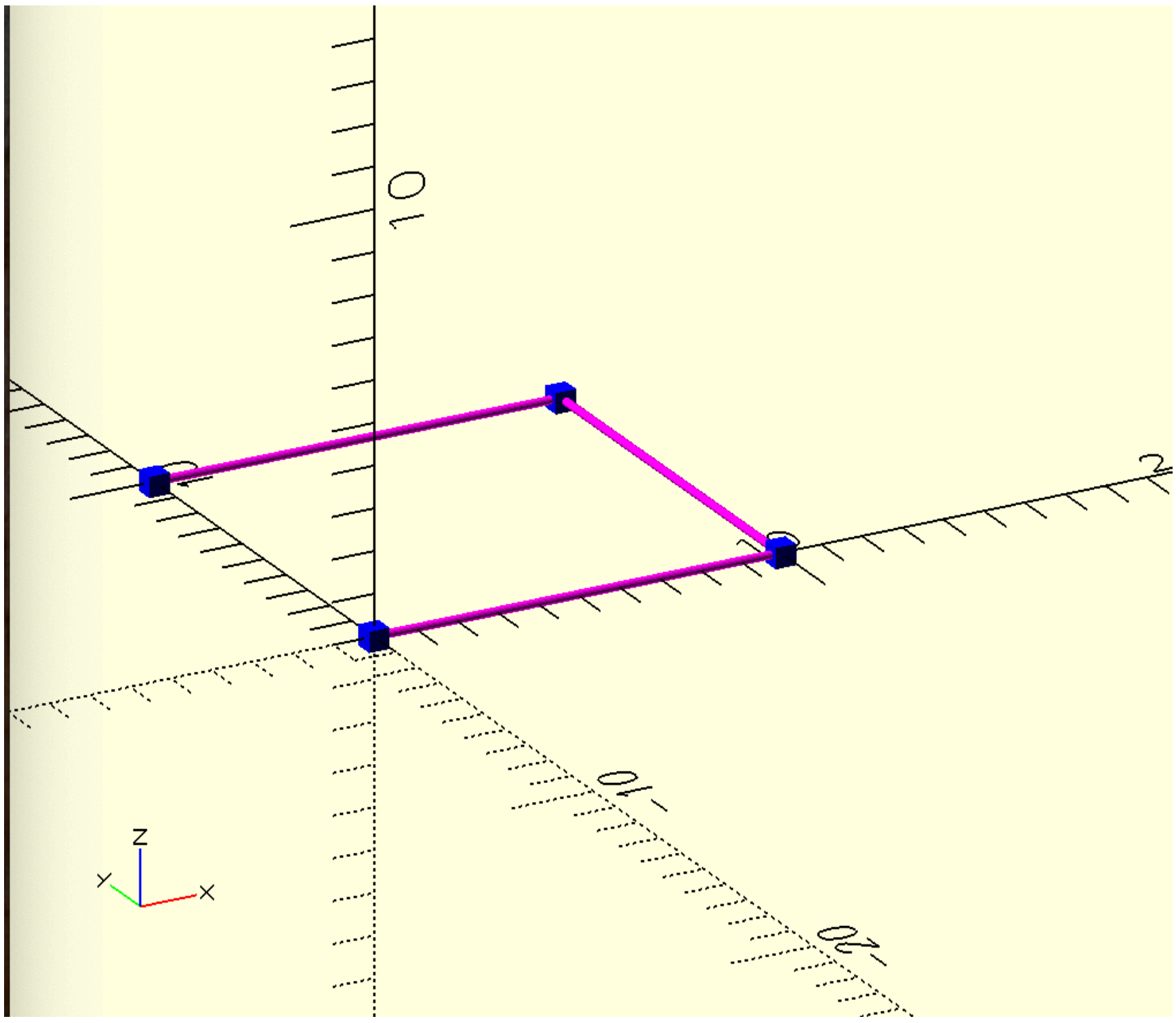
Lines

In [6]:
```
l1=[[10,0,0],[10,0,10]]
fileopen(f'''
color("blue") points({l1},.5);

// p_line3d module is used for showing lines or polylines
// in this example line "l1" of diameter 0.2 mm is shown

color("magenta") p_line3d({l1},.2);
''')
```



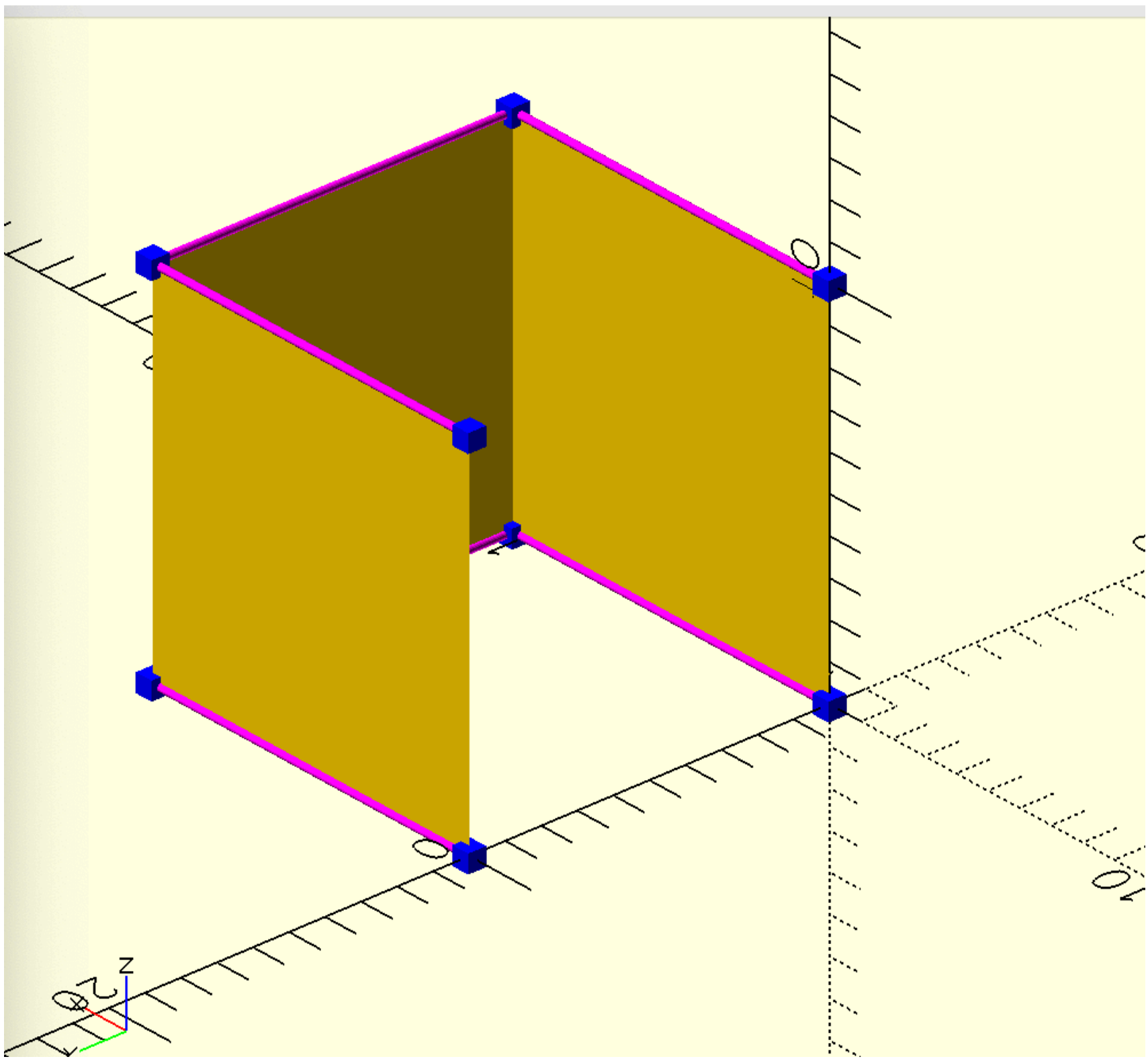
Polylines

In [ ]:
```
l2=cr2dt([[0,0],[10,0],[0,10],[-10,0]])
fileopen(f'''
color("blue") points({l2},.5);
color("magenta") p_line3d({l2},.2);

''')
```



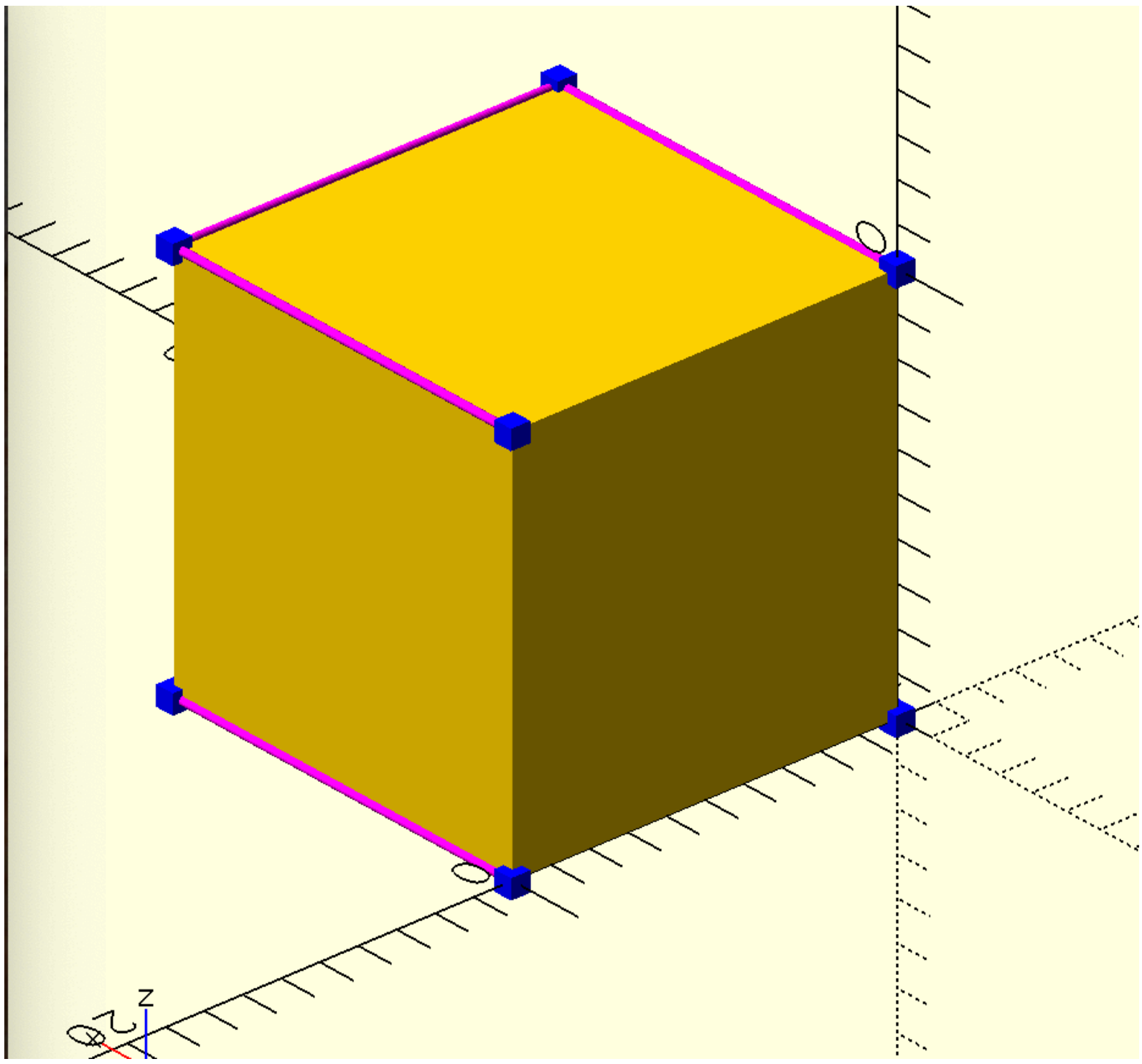
Surface

In [ ]:
```
l2=cr2dt([[0,0],[10,0],[0,10],[-10,0]])
s1=linear_extrude(l2,10)
fileopen(f'''
color("blue") for(p={s1}) points(p,.5);
color("magenta")for(p={s1}) p_line3d(p,.2);

// pay attention to the swp_surf module here
// swp_surf shows the surface covered by the polylines and is very important
// to understand as intersections are calculated based on intersecting surfaces

{swp_surf(s1)}

''')
```



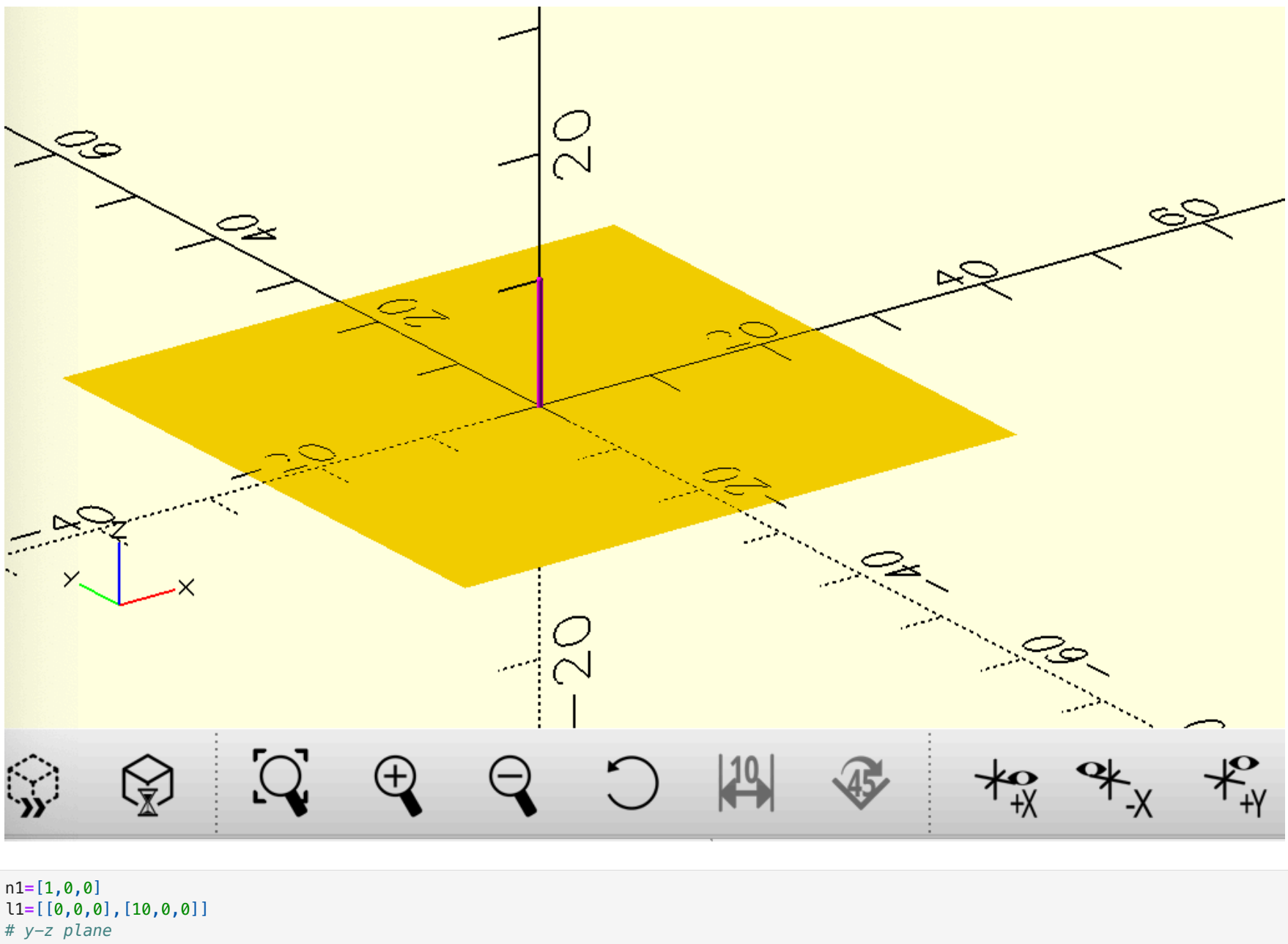
Solid

In [ ]:
```
l2=cr2dt([[0,0],[10,0],[0,10],[-10,0]])
s1=linear_extrude(l2,10)
fileopen(f'''
color("blue") for(p={s1}) points(p,.5);
color("magenta")for(p={s1}) p_line3d(p,.2);
{swp(s1)}

''')
```



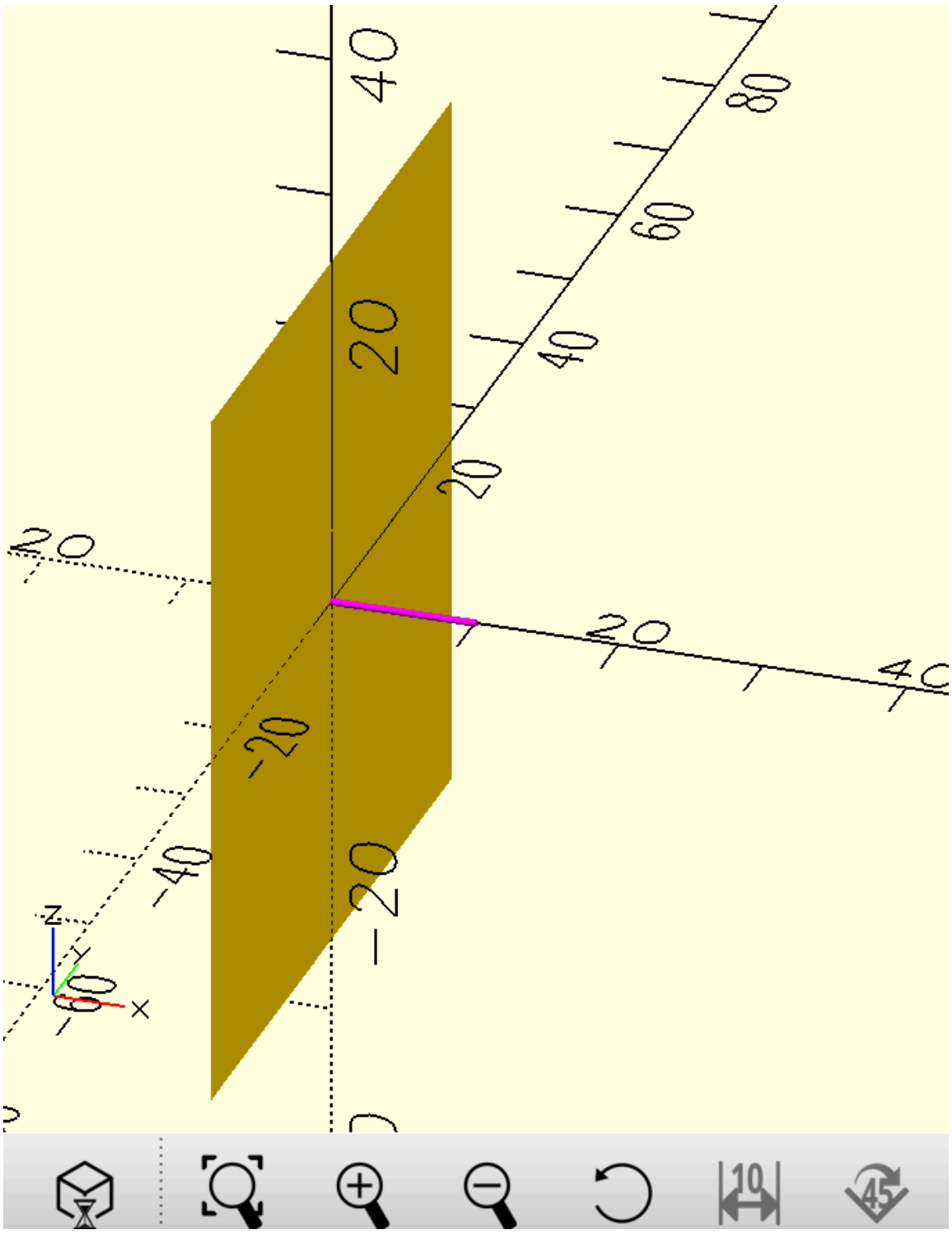
Planes

In [ ]:
```
n1=[0,0,1]
l1=[[0,0,0],[0,0,10]]
# x-y plane
pl1=plane(n1,size=[50,50], intercept=[0,0,0])
fileopen(f'''
color("magenta") p_line3d({l1},.5);
{swp_surf(pl1)}
''')
```

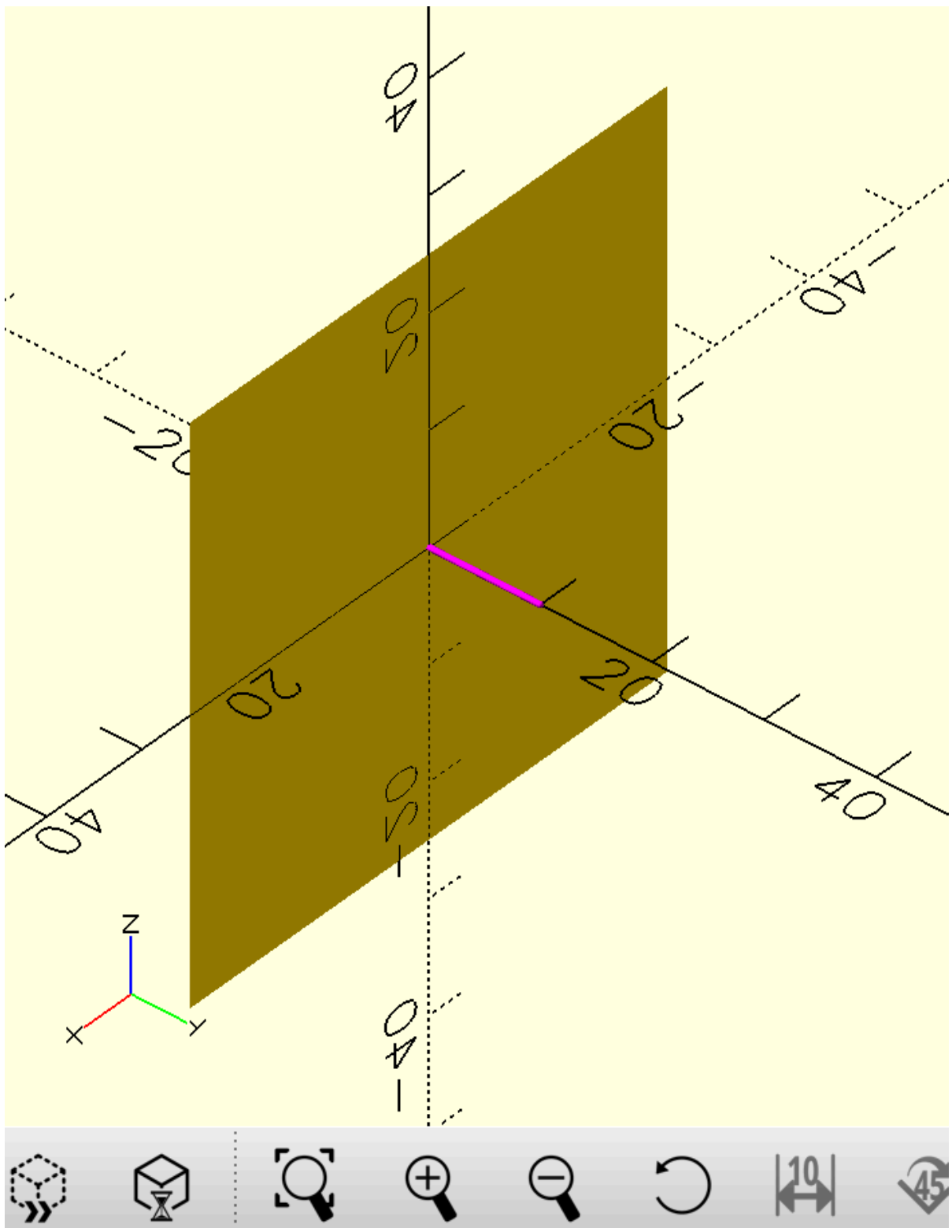


In [ ]:
```
n1=[1,0,0]
l1=[[0,0,0],[10,0,0]]
# y-z plane
```

```
pl1=plane(n1,size=[50,50], intercept=[0,0,0])
fileopen(f'''
color("magenta") p_line3d({l1},.5);
{swp_surf(pl1)}
''')
```

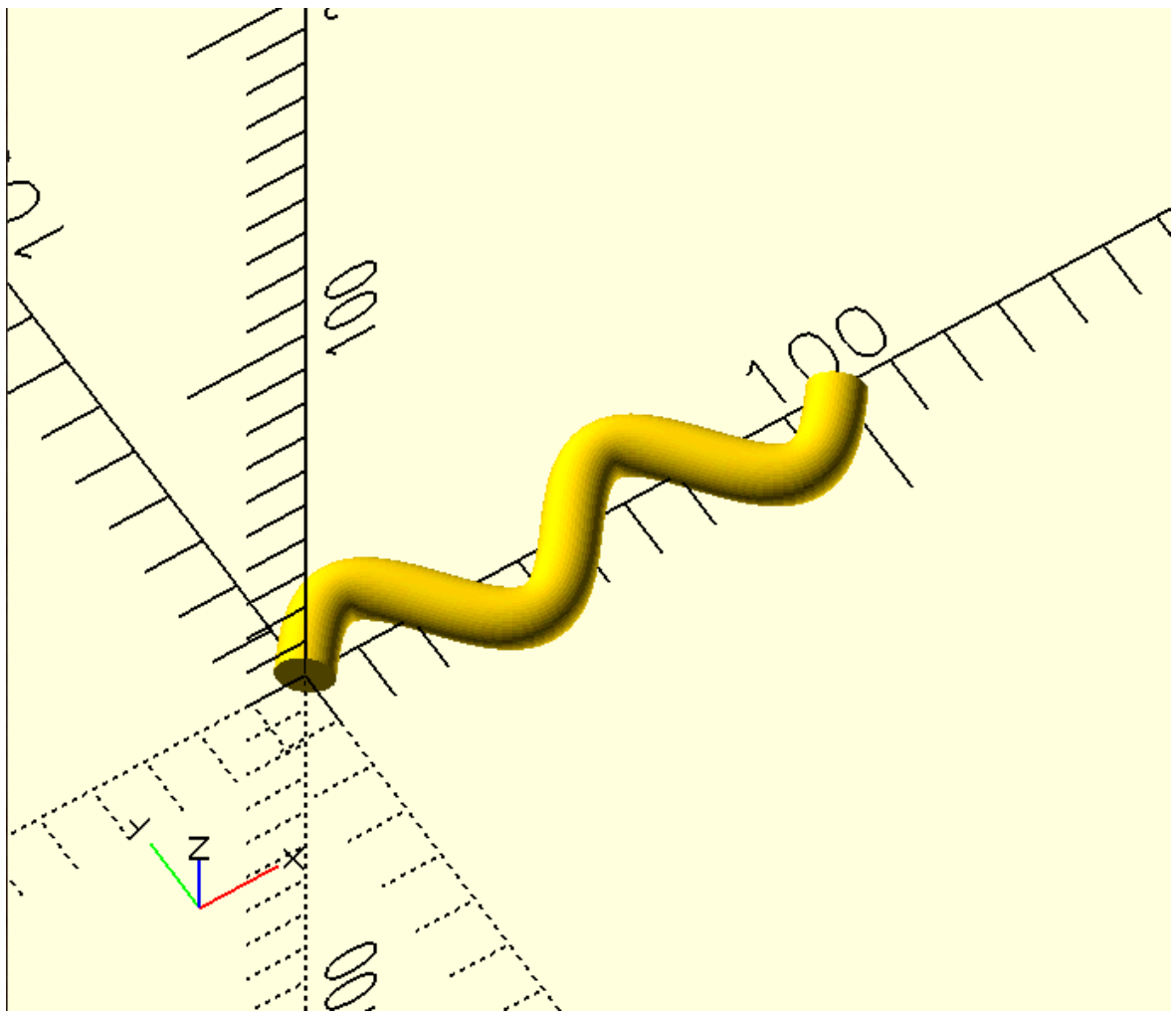


In [ ]:
```
n1=[0,1,0]
l1=[[0,0,0],[0,10,0]]
# x-z plane
pl1=plane(n1,size=[50,50], intercept=[0,0,0])
fileopen(f'''
color("magenta") p_line3d({l1},.5);
{swp_surf(pl1)}
''')
```



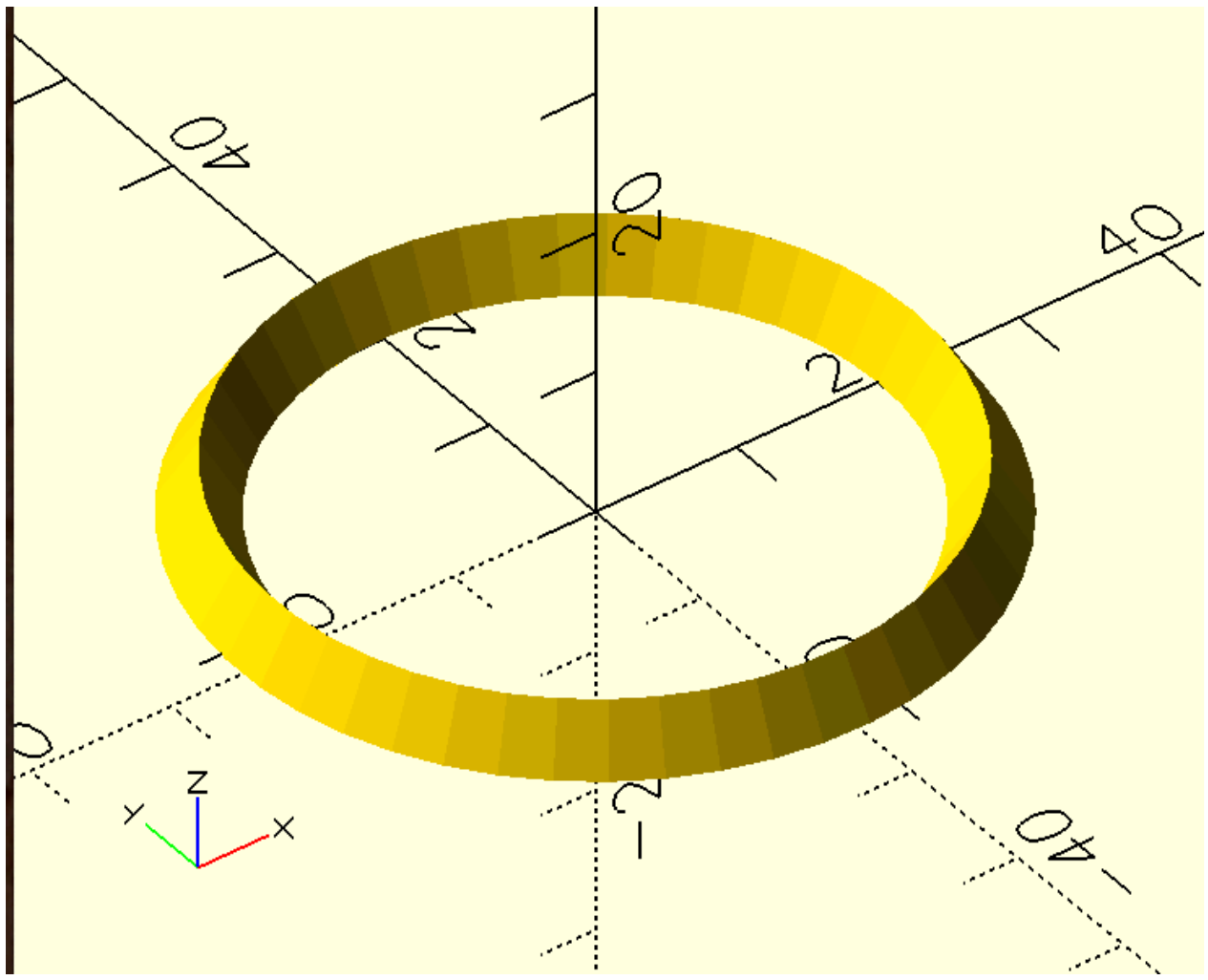
Extrude along path

In [ ]:
```
# circular section extruded along open path
sec=circle(5)
path=c23(sinewave(l=100,n=2,a=10,p=100))
sol=path_extrude_open(sec,path)
fileopen(f'''
{swp(sol)}
''')
```



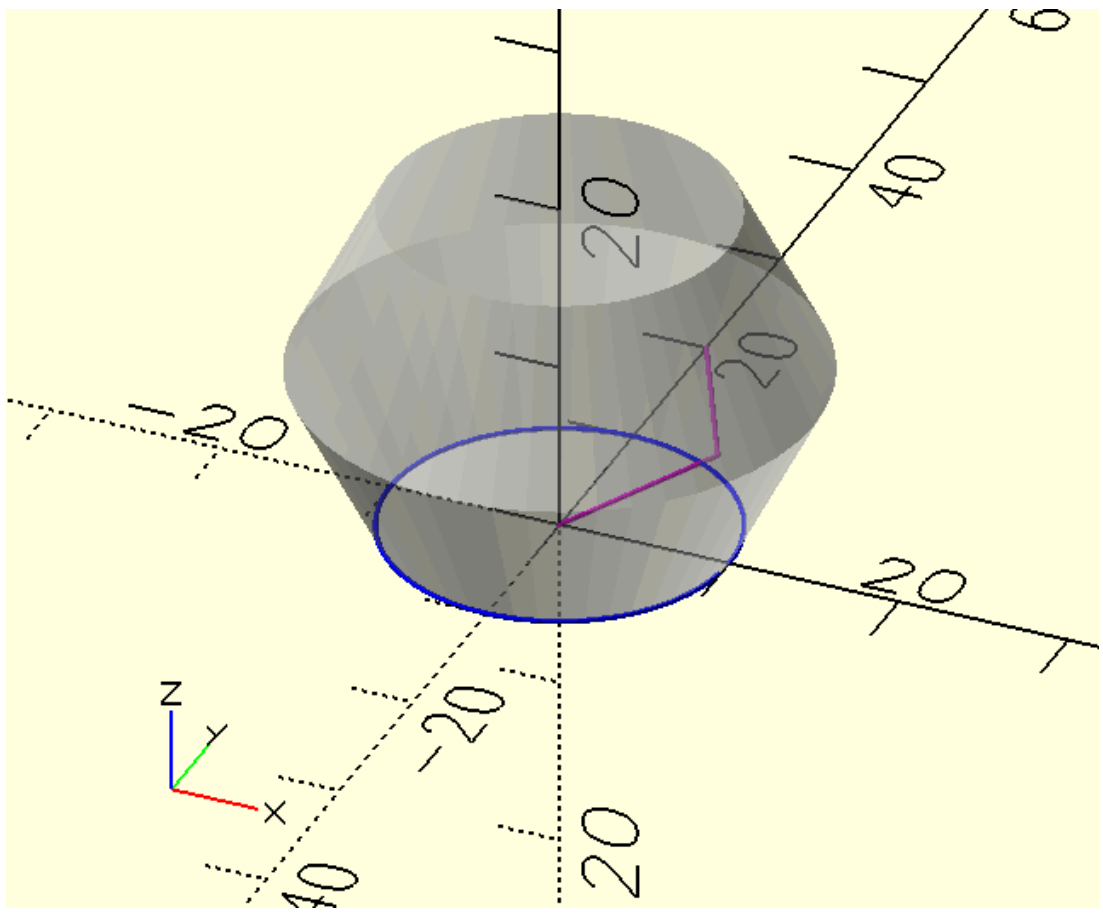
In [ ]:
```
# triangular section extruded along closed path
sec=[[0,0],[5,0],[2.5,4]]
path=c23(circle(20))
sol=path_extrude_closed(sec,path)
fileopen(f'''
// pay attention to the swp_c module here
```

```
// swp_c is to be used where the loop is closing like the way here
{swp_c(sol)}
''')
```



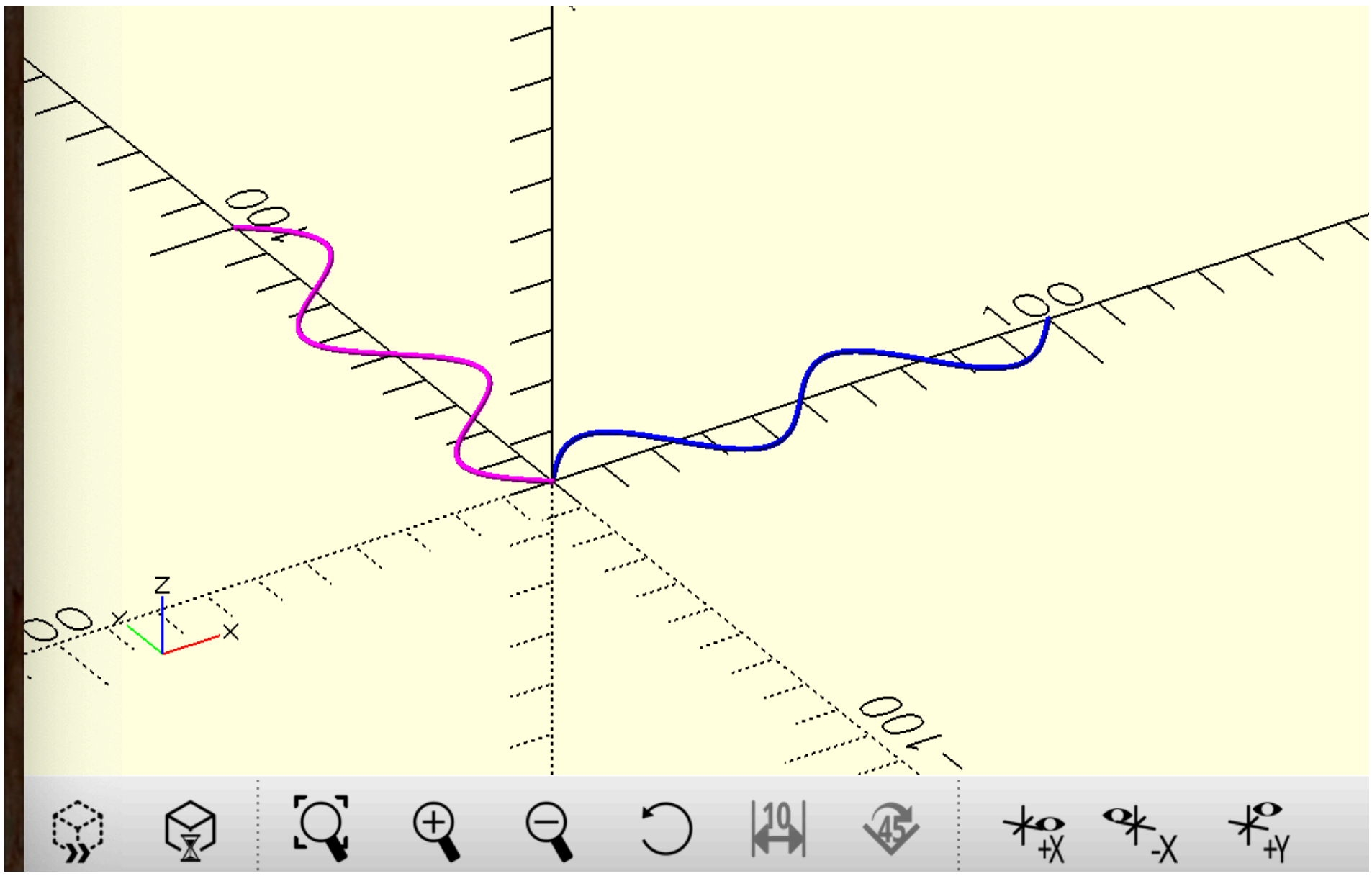
Sculpting along path

In [ ]:
```
sec=circle(10)
path=[[0,0],[5,10],[0,20]] # x-coordinates work as offset and y-coordinates work as z-translate of sec
sol=prism(sec,path)
fileopen(f'''
color("blue") p_line3d({sec},.3);
color("magenta") p_line3d({path},.3);
%{swp(sol)}

''')
```

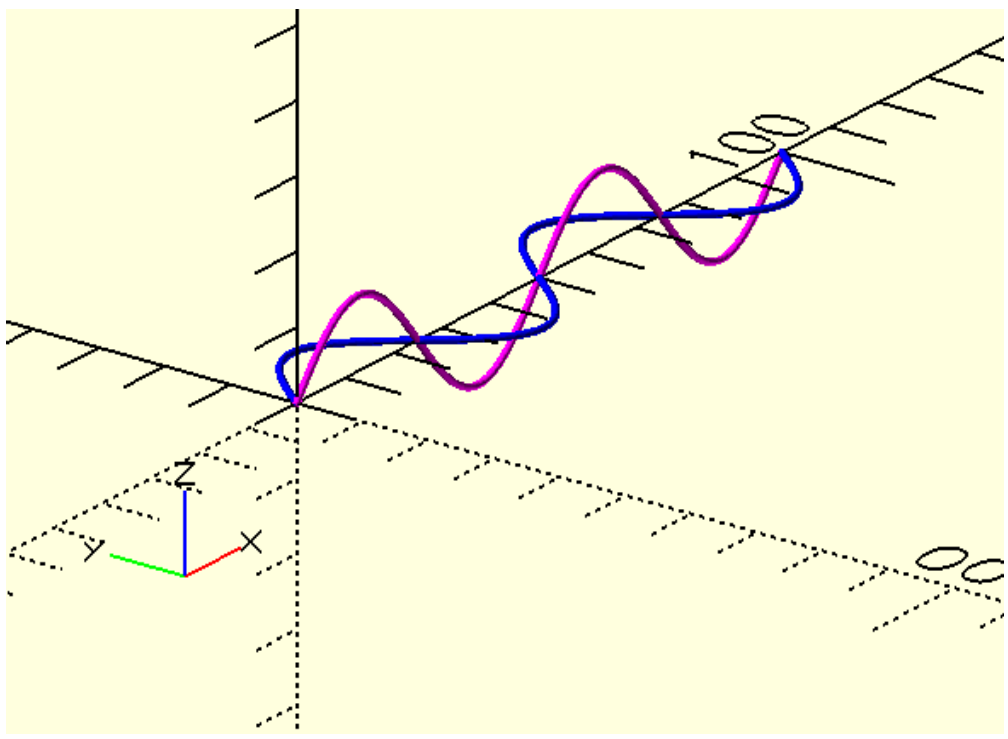


Rotation : Right hand thumb-rule (if thumb is pointed in the direction of axis, fingers curled in the direction of rotation)

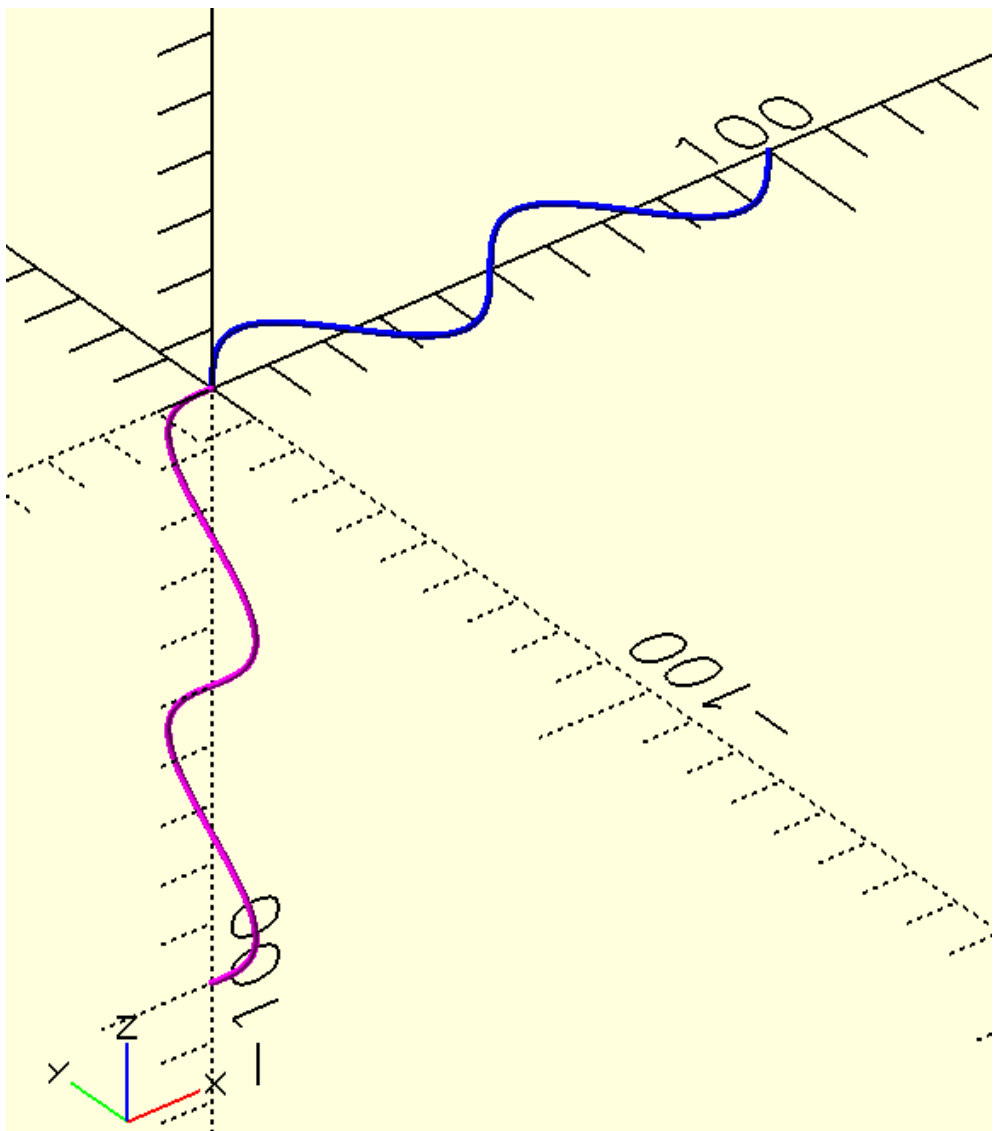
In [ ]:
```
l1=sinewave(100,2,10,100)
l2=rot('z90',l1) # l1 rotated by 90 deg along z-axis
fileopen(f'''
// original line 'l1'
color("blue") p_line3d({l1},1);
//rotated line
color("magenta") p_line3d({l2},1);

''')
```



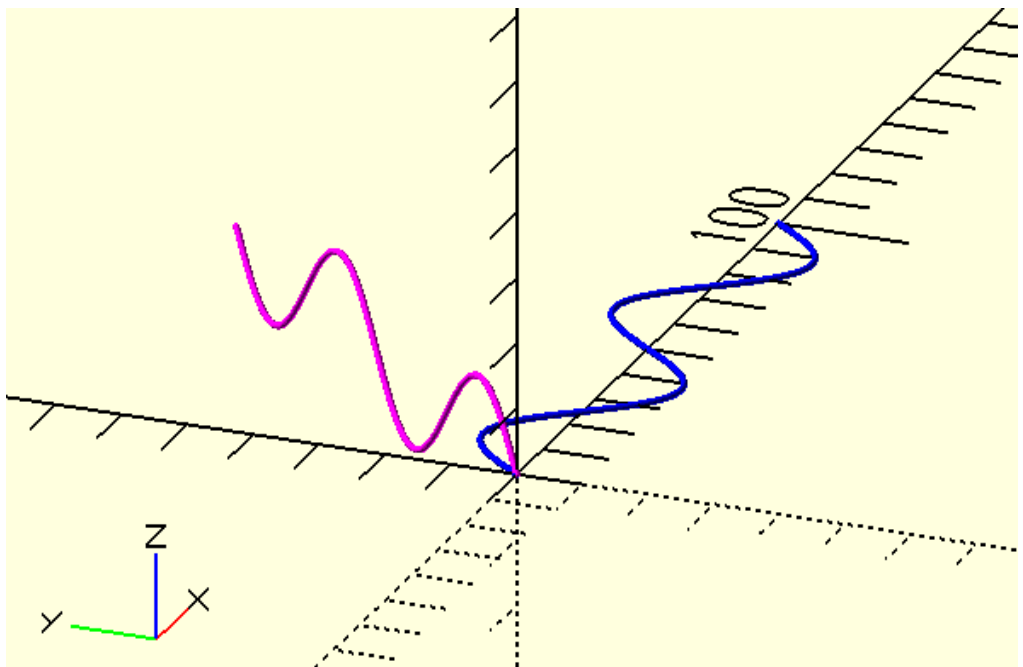
In [ ]:
```
l1=sinewave(100,2,10,100)
l2=rot('x90',l1) # l1 rotated by 90 deg along x-axis
fileopen(f'''
// original line 'l1'
color("blue") p_line3d({l1},1);
//rotated line
color("magenta") p_line3d({l2},1);

''')
```



In [ ]:
```
l1=sinewave(100,2,10,100)
l2=rot('y90',l1) # l1 rotated by 90 deg along y-axis
fileopen(f'''
// original line 'l1'
color("blue") p_line3d({l1},1);
//rotated line
color("magenta") p_line3d({l2},1);

''')
```

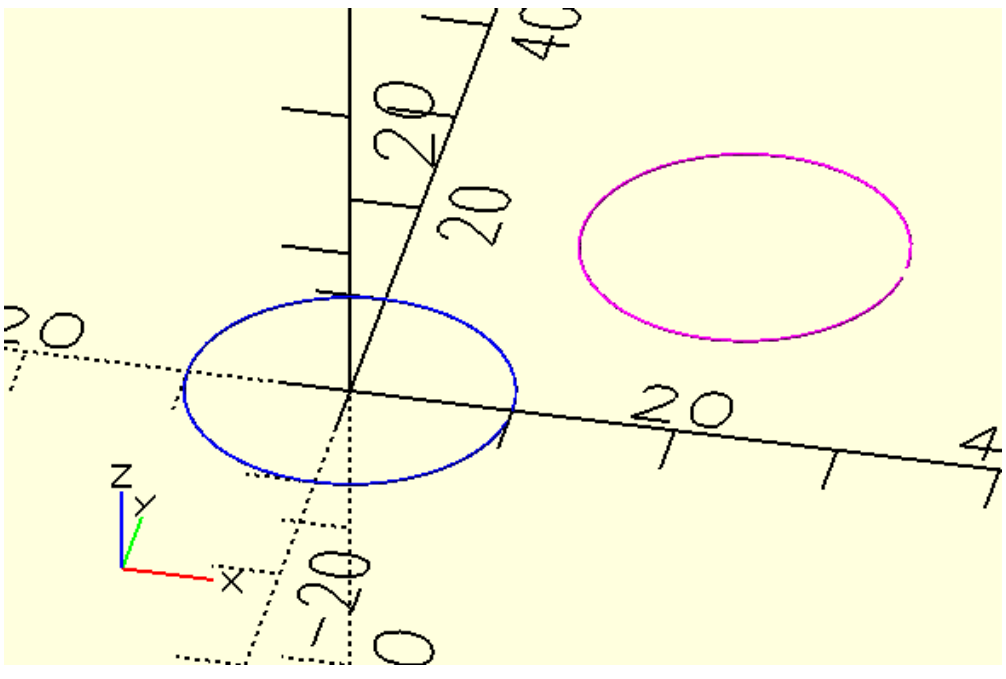


In [ ]:
```
l1=sinewave(100,2,10,100)
l2=rot('x90z45',l1) # multiple rotation of l1 (rotated by 90 deg along x-axis
# and then 45 deg along z-axis
fileopen(f'''
// original line 'l1'
color("blue") p_line3d({l1},1);
//rotated line
color("magenta") p_line3d({l2},1);

''')
```

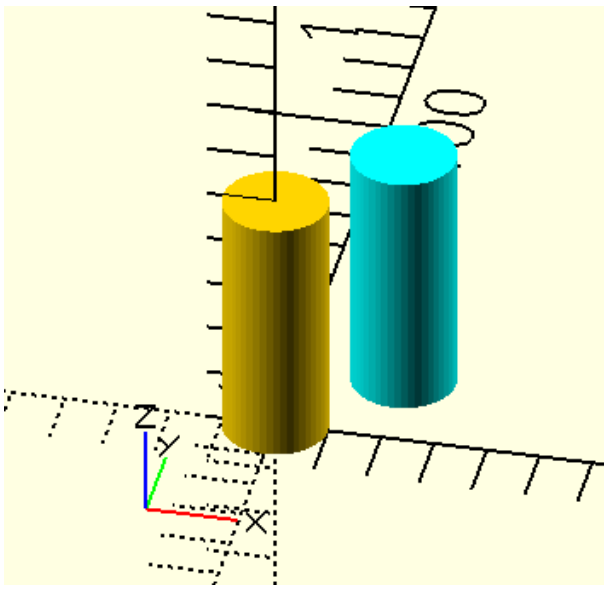


Translate: are of 2d and 3d type

In [8]:
```
# example of translate in 2 d coordinates

c1=circle(10)
c2=translate_2d([20,20],c1)
fileopen(f'''
// original circle
color("blue") p_line3d({c1},.2);

// translated circle
color("magenta") p_line3d({c2},.2);

''')
```

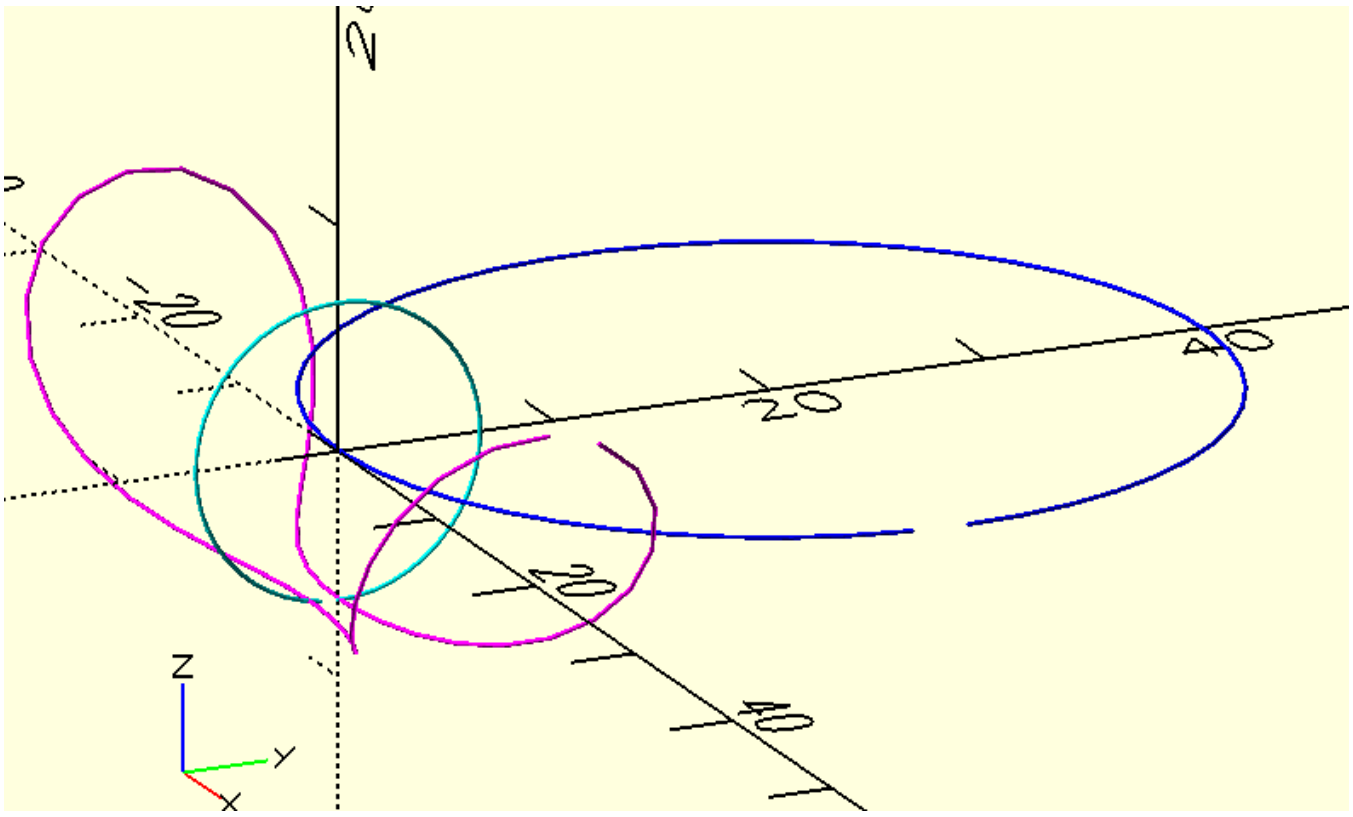


In [7]:
```
# example of translate in 3d coordinate
c1=linear_extrude(circle(10),50)
c2=translate([20,20,0],c1)
fileopen(f'''
// original cyclinder
{swp(c1)}
// translated cylinder by vector [20,20,0]
color("cyan"){swp(c2)}

''')
```



wrap around a section over a path

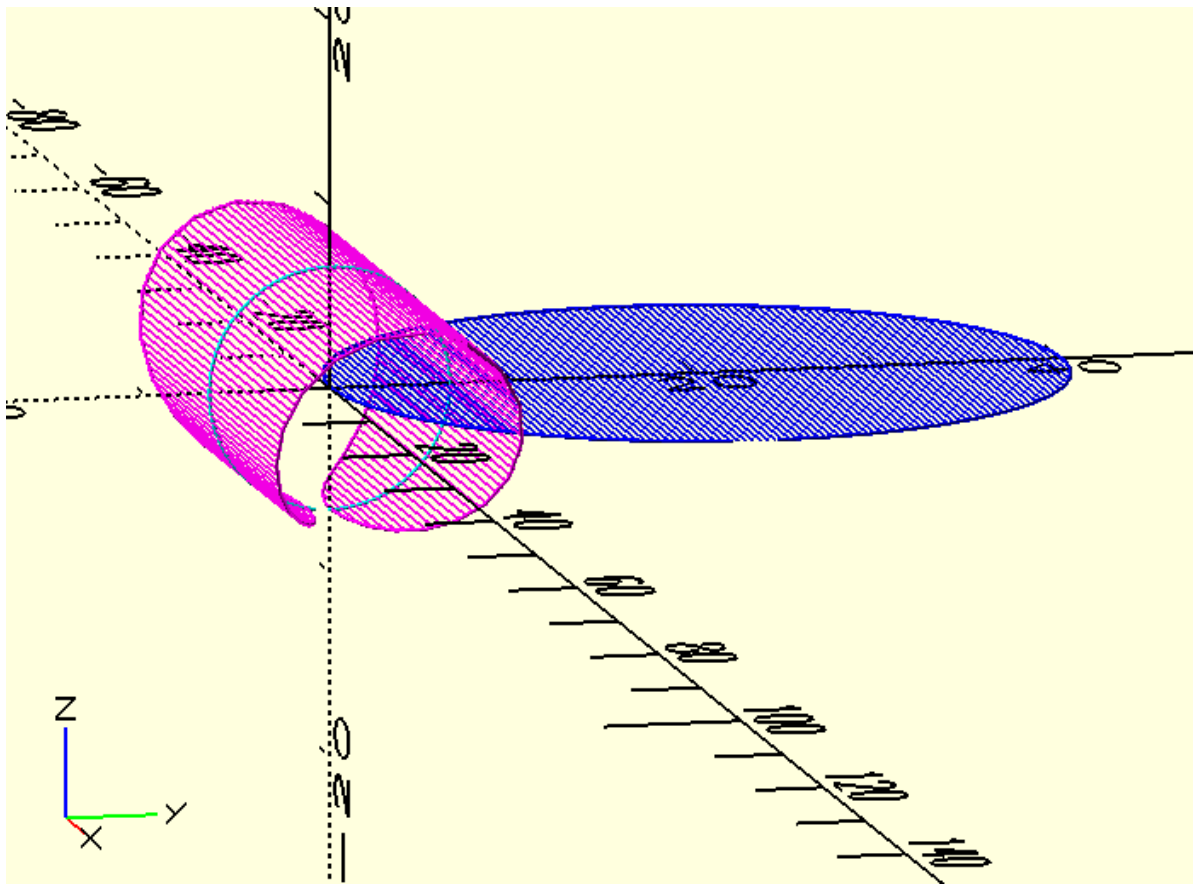
In [20]:
```
c1=translate([0,20.1,0],circle(20))
path=rot('y90',circle(40.2/(2*pi)+.2))
c2=wrap_around(c1,path)

fileopen(f'''
color("blue") p_line3d({c1},.2);
color("cyan") p_line3d({path},.2);
color("magenta") p_line3d({c2},.2);

''')
```



wrap around a surface over a path

In [19]:
```
c1=translate_2d([0,20.1],circle(20))
s1=h_lines_sec(c1,100)
path=rot('y90',circle(40.2/(2*pi)+.2))
c2=wrap_around(c1,path)
s2=[wrap_around(p,path) for p in s1]
```

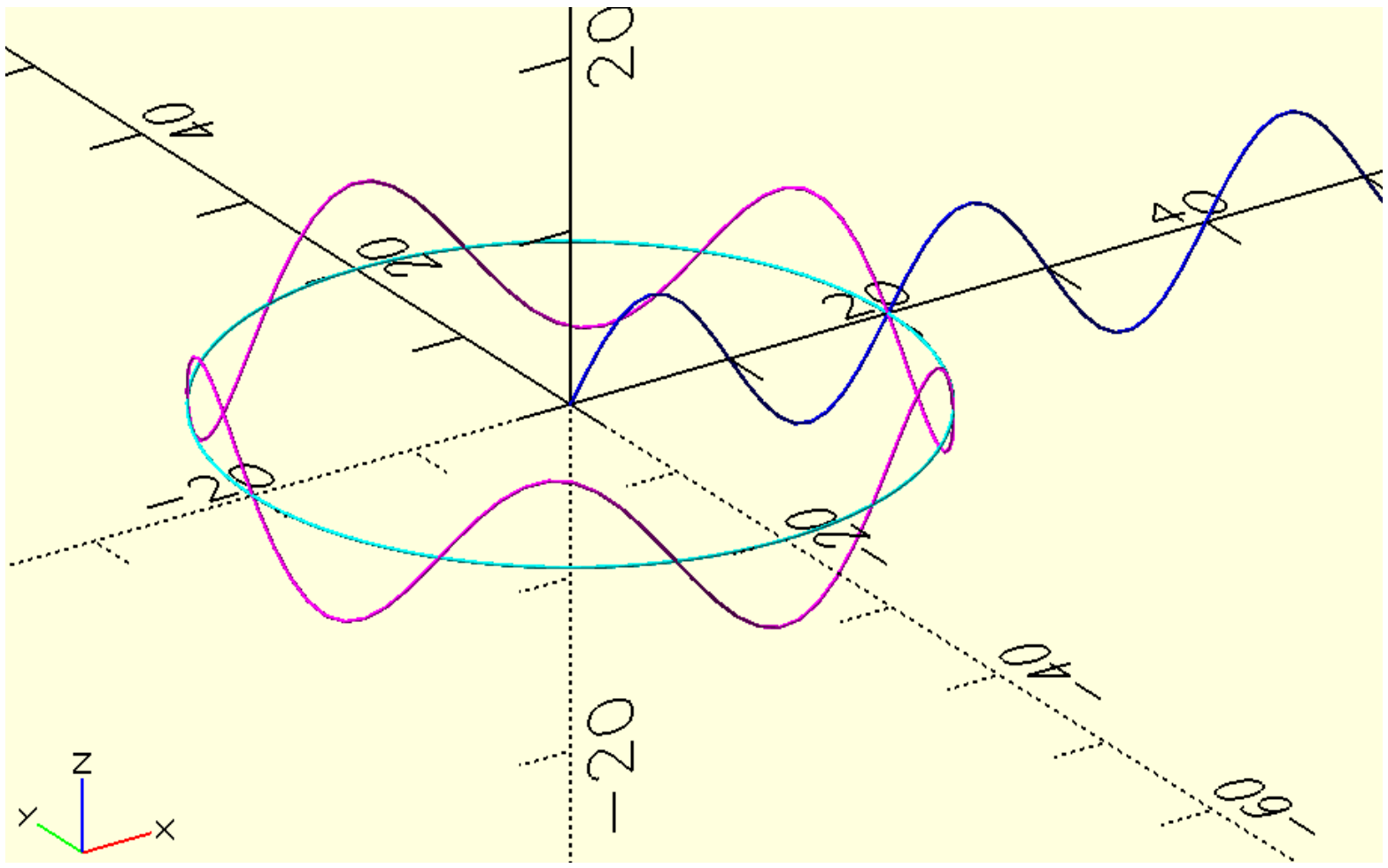
```
fileopen(f'''
color("blue") p_line3d({c1},.2);
color("cyan") p_line3d({path},.2);
color("magenta") p_line3d({c2},.2);

color("blue") for(p={s1}) p_line3d(p,.1,1);
color("magenta") for(p={s2}) p_line3d(p,.1,1);
''')
```



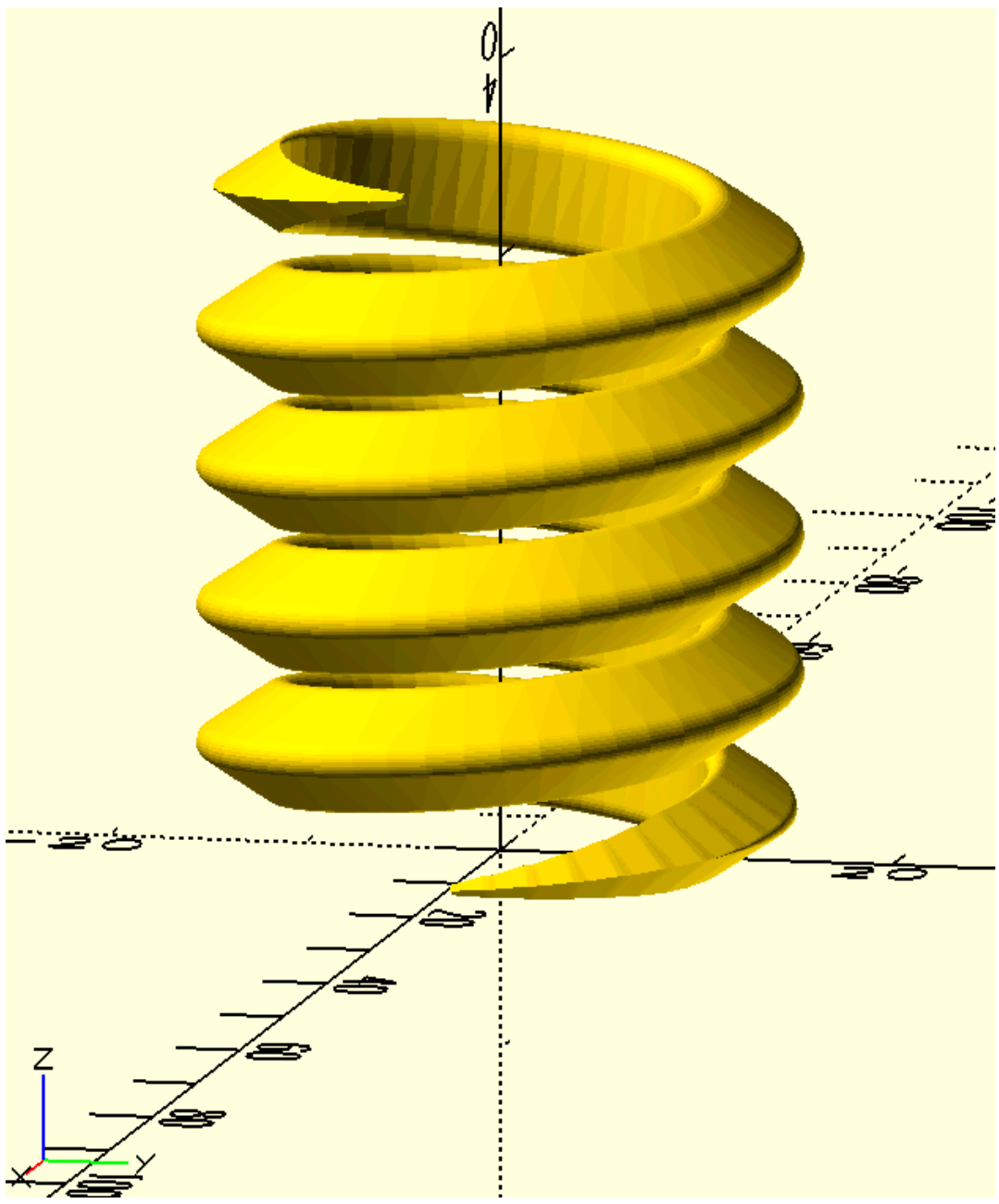
other methods of wrapping a polyline/ solid around a path

In [39]:
```
c1=rot('x90',sinewave(100,5,5,100))
path=c23(arc(20,0,360,s=99))
c2=extrude_wave2path(c1,path)

fileopen(f'''
color("blue") p_line3d({c1},.2);
color("cyan") p_line3d({path},.2);
color("magenta") p_line3dc({c2},.2);

''')
```

Out[39]: (100, 100)



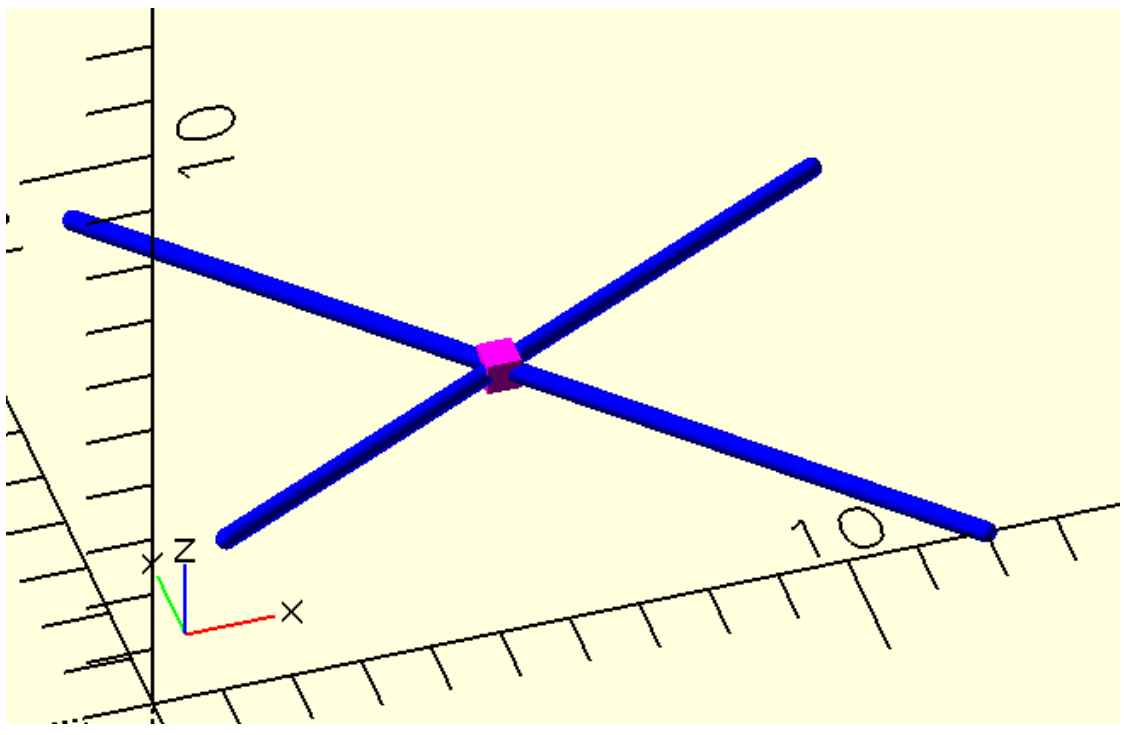
In [59]:
```
c1=rot2d(-90,cr2dt([[-4,0,1],[8,0,1],[-4,6,1]],10))
path=m_points1_o(cr2dt([[-2,0],[2,4,5],[0,50,5],[-2,4]],10),200,.01)
sol=prism(c1,path)
path1=helix(10,7,5,10)
path1=path2path1(path,path1)
sol1=sol2path(sol,path1)
fileopen(f'''
{swp(sol1)}
//color("blue") p_line3d({path},.5,1);
''')
```



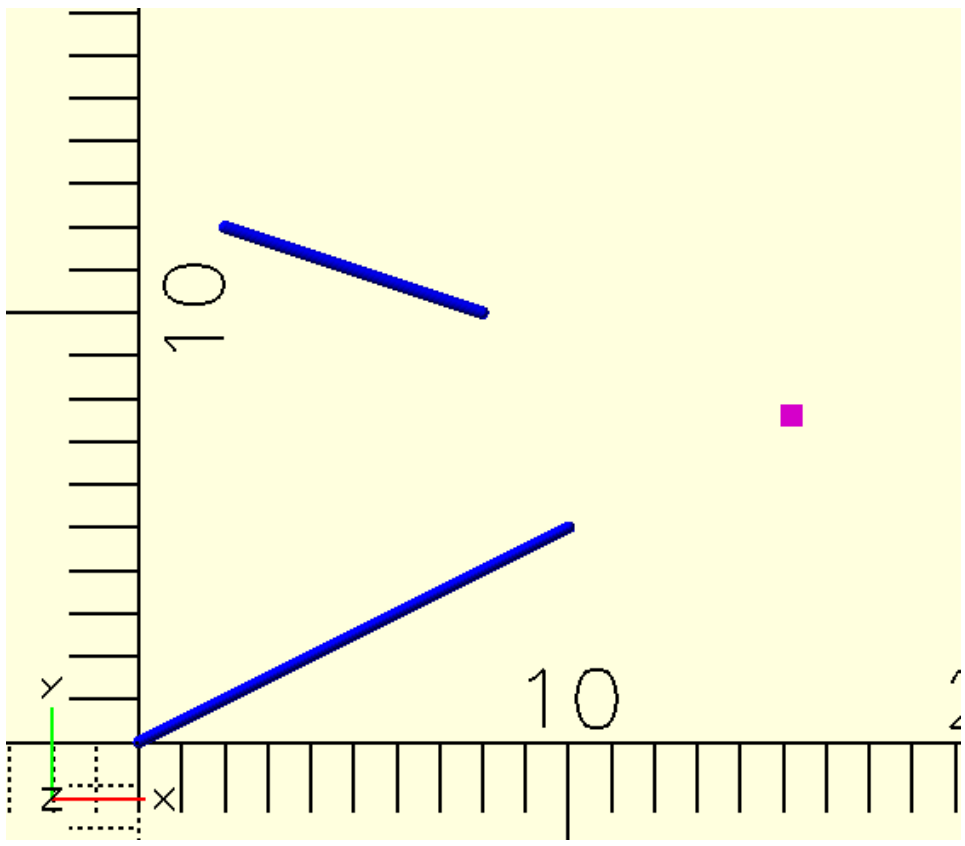
Intersections

intersection between line to line (2d)

In [63]:
```
l1=point_vector([2,3],[10,5])
l2=point_vector([2,10],[10,-10])
p0=s_int1([l1,l2])[0]
fileopen(f'''
color("blue") for(p={[l1,l2]}) p_line3d(p,.3);
color("magenta") points({[p0]},.5);
''')
```

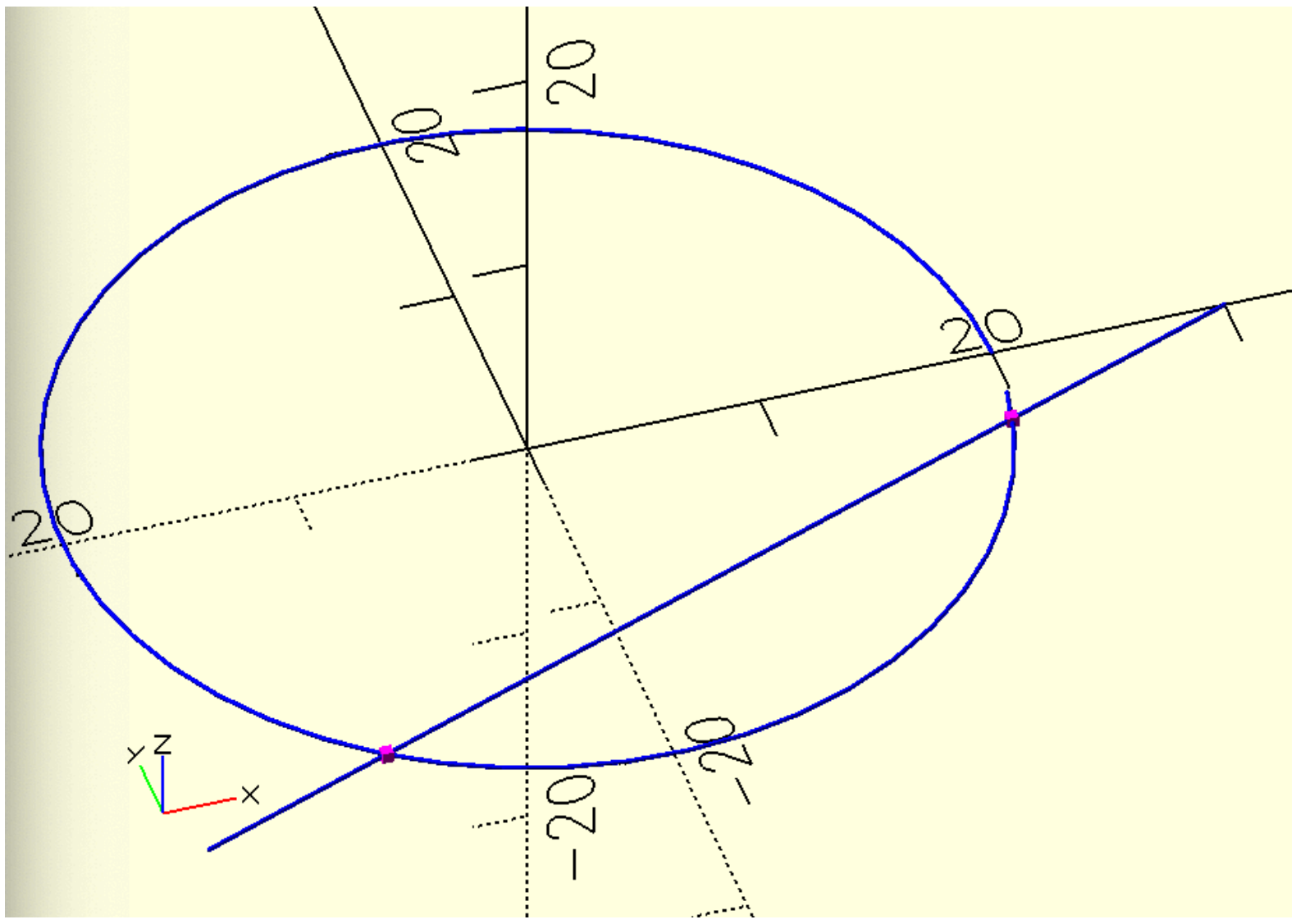


In [93]:
```
# intersection point between 2 lines even if they are not directly intersecting
l1=[[0,0],[10,5]]
l2=[[2,12],[8,10]]
p0=i_p2d(l1,l2)
fileopen(f'''
color("blue") for(p={[l1,l2]}) p_line3d(p,.3);
color("magenta") points({[p0]},.5);

''')
```



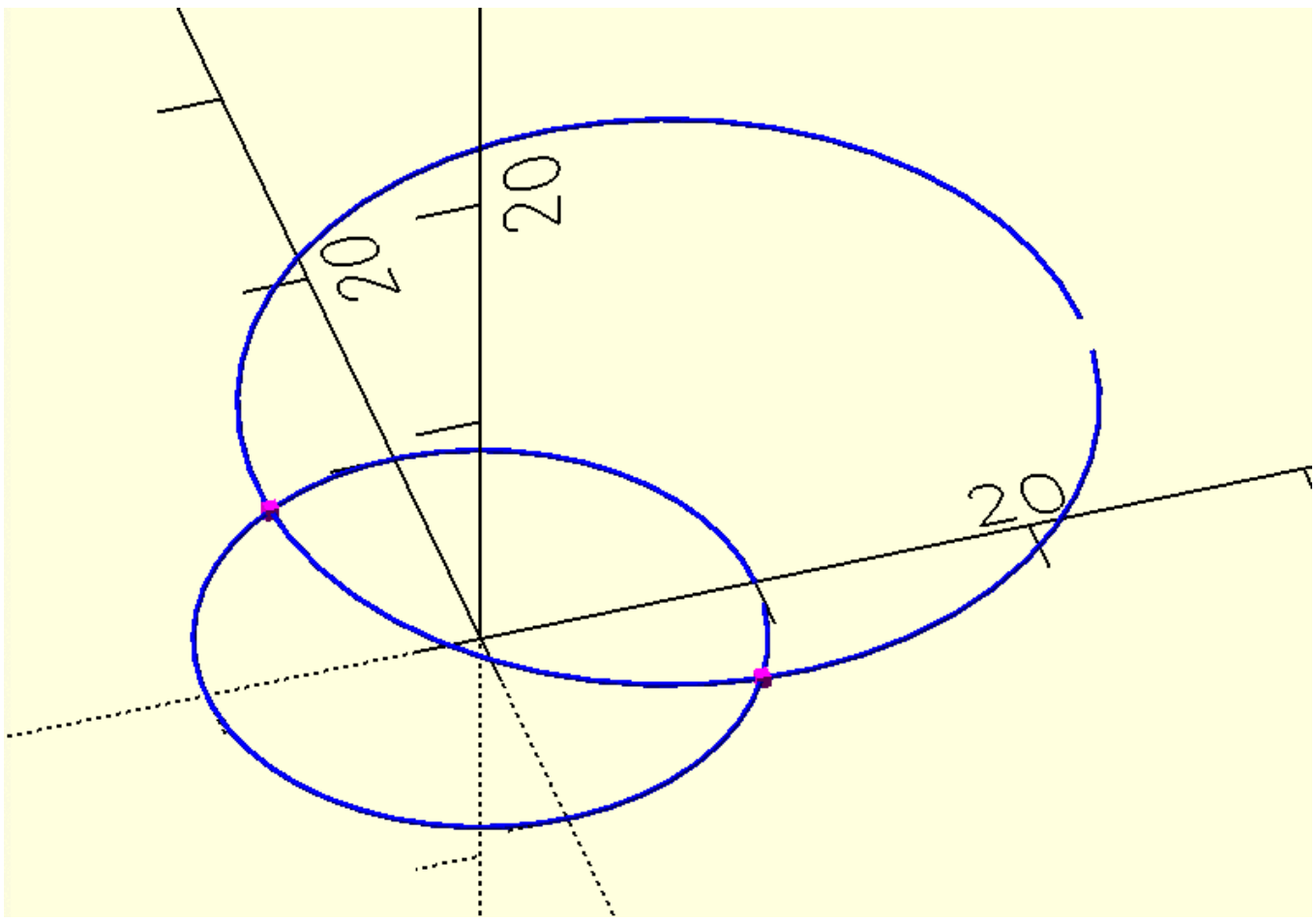
intersection between a polyline and line

In [65]:
```
c1=circle(20)
l1=point_vector([-20,-20],[50,20])
p0=s_int1([l1]+seg(c1))
fileopen(f'''
color("blue") for(p={[l1,c1]}) p_line3d(p,.2);
color("magenta") points({p0},.5);

''')
```

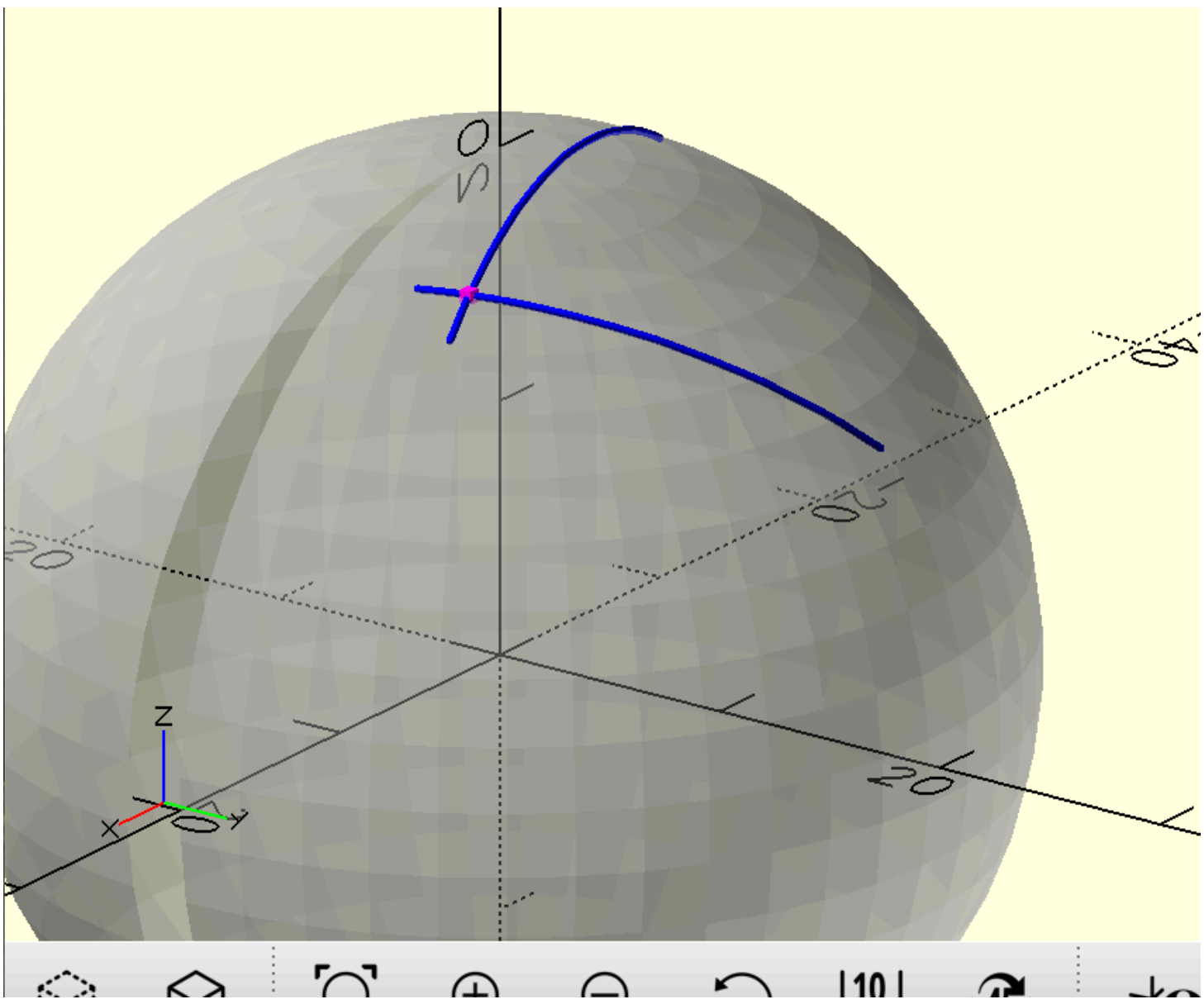


intersection between 2 polylines

In [70]:
```
c1=circle(10)
c2=circle(15,[10,10])
p0=s_int1(seg(c1)+seg(c2))
fileopen(f'''
color("blue") for(p={[c1,c2]}) p_line3d(p,.2);
color("magenta")points({p0},.5);
''')
```



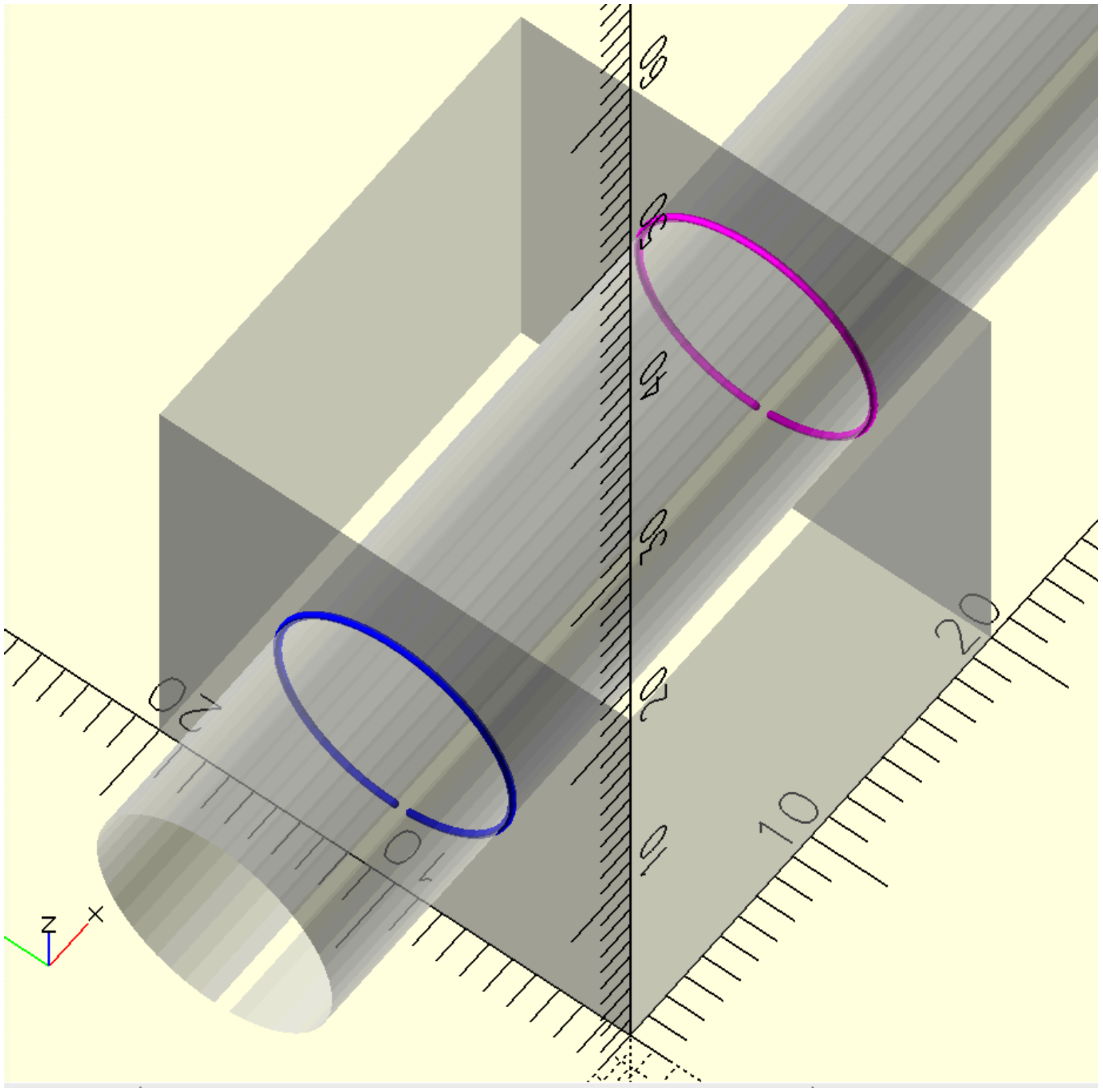
In [119…
```
# intersection between 2 polylines in 3d space
s1=sphere(20)
l1=c23(homogenise([[-10,0],[10,5]],1))
l1=plos(s1,l1,[0,0,1])
l2=c23(homogenise([[0,-15,0],[-7,5,0]],1))
l2=plos(s1,l2,[1,2,2])
p0=s_int1_3d(seg(l1)+seg(l2))[0]
fileopen(f'''
%{swp_surf(s1)}
color("blue") for(p={[l1,l2]}) p_line3d(p,.3);
color("magenta") points({[p0]},.5);
''')
```



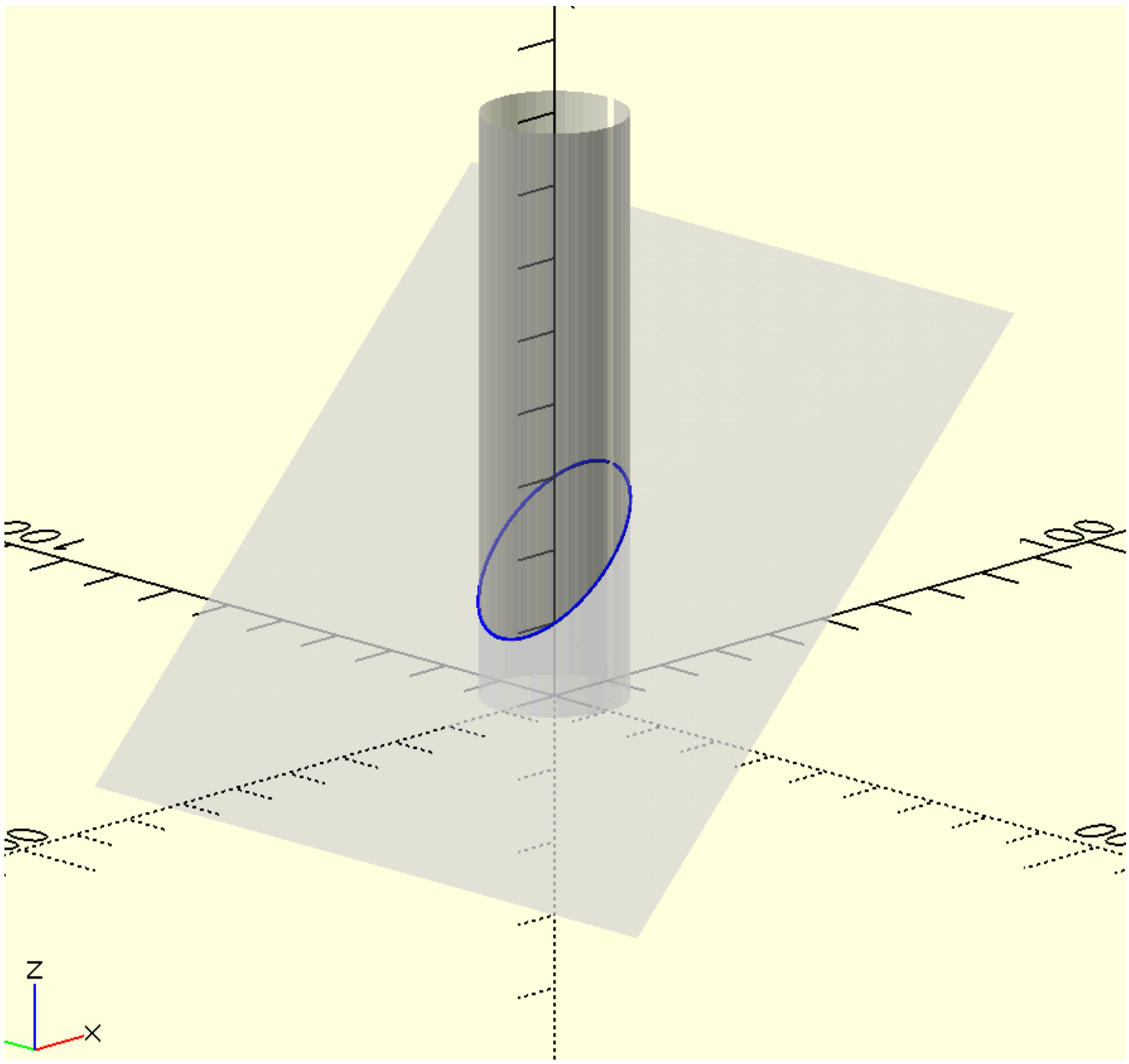
intersection between 2 surfaces

In [96]:
```
s1=linear_extrude(square(20),20)
s2=translate([-10,10,10],rot('y90',linear_extrude(circle(5),50)))
l1=ip_sol2sol(s1,s2,n=-1)
l2=ip_sol2sol(s1,s2,n=0)

fileopen(f'''
%{swp_c(s1)}
%{swp_surf(s2)}
color("blue") p_line3d({l1},.3);
color("magenta") p_line3d({l2},.3);
''')
# Note: To debug issues related to intersection:
# There are 2 surfaces surface1 (s1 in this case) and surface2(s2 in this case)
# surface 1 is intersected by surface 2
```

```
# So surface1 should be rendered with module "swp_c"
# surface2 should be rendered with module "swp_surf"
```



In [88]:
```
pl1=plane([-1,0,1],[100,100],[0,0,20])
c1=cylinder(r=10,h=80)
p0=ip_sol2sol(pl1,c1)
fileopen(f'''
%{swp_c(pl1)}
%{swp_surf(c1)}
color("blue") p_line3d({p0},.5);
''')
```


20
100
0
0
Z
X